Instituto Tecnológico de Costa Rica

Escuela de Ingeniería Electrónica

Desarrollo de un módulo de decodificación de fila para una memoria SRAM de 2 kb, en un proceso CMOS de 180 nm

Informe de Proyecto de Graduación para optar por el título de Ingeniero en Electrónica con el grado académico de Licenciatura

Bernardo Emilio Rodríguez Hall

Cartago, Costa Rica

26 de noviembre de 2018

INSTITUTO TECNOLÓGICO DE COSTA RICA

ESCUELA DE INGENIERÍA ELECTRÓNICA

PROYECTO DE GRADUACIÓN

ACTA DE APROBACIÓN

Defensa de Proyecto de Graduación
Requisito para optar por el título de Ingeniero en Electrónica
Grado Académico de Licenciatura
Instituto Tecnológico de Costa Rica

El Tribunal Evaluador aprueba la defensa del proyecto de graduación denominado Desarrollo de un módulo de decodificación de fila para una memoria SRAM de 2kb, en un proceso CMOS de 180 nm, realizado por el señor Bernardo Emilio Rodríguez Hall y, hace constar que cumple con las normas establecidas por la Escuela de Ingeniería Electrónica del Instituto Tecnológico de Costa Rica.

Miembros del Tribunal Evaluador

Ing. Aníbal Ruiz Barquero

Profesor lector

Ing. Luis C. Rosales Alpízar

Profesor lector

Dr.-Ing. Alfonso Chacón Rodríguez

Profesor asesor

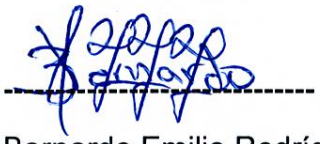
Cartago, 23 de Noviembre del 2018

Declaración de autenticidad

Declaro que el presente Proyecto de Graduación ha sido realizado, en su totalidad, por mi persona, utilizando y aplicando literatura referente al tema e introduciendo conocimientos propios.

En los casos en que he utilizado material bibliográfico, he procedido a indicar las fuentes mediante citas.

En consecuencia, asumo la responsabilidad total por el trabajo de graduación realizado y por el contenido del correspondiente informe final.



Bernardo Emilio Rodríguez Hall

Cedula: 207450471

Resumen

En el del Instituto Tecnológico de Costa Rica, en la Escuela de Ingeniería en Electrónica, el Laboratorio de Circuitos Integrados (DCILab) actualmente desarrolla una microaquitectura ISA RISC-V 32 E, orientado para dispositivos médicos implantables con procesamiento avanzado. Dentro de esta microaquitectura en la unidad de manejo de memoria, se desarrolló como parte de la memoria, el decodificador de acceso y sus periféricos, de modo que las diferentes celdas que lo constituyen funcionen como celdas estándar como parte de una biblioteca y así crear un generador de memoria automático propio del laboratorio.

Palabras clave: SRAM, Decodificador, Buffer, Latch.

Abstract

At the Instituto Tecnológico de Costa Rica, at the Escuela de Ingeniería Electrónica, the DCILab (Laboratorio de Diseño de Circuitos Integrados), is currently developing an ISA RISC-V 32 E microarchitecture, oriented to implantable medic devices with advanced processing. It was developed the access decoder and its peripherals of the memory, part of the MMU in this microarchitecture. The objective is to build different cells that function as a standard part of a library and then create an automatic memory generator of the laboratory itself in future projects.

Keywords: SRAM, Decoder, Buffer, Latch.

Agradecimientos

Quiero agradecer en primer lugar a mi madre y mis hermanos, ellos siempre me han apoyado y ayudado en el camino hacia mi superación personal y profesional en todo momento.

También quiero agradecer a los profesores del Instituto Tecnológico de Costa Rica, que estuvieron presentes a lo largo de mi carrera por su paciencia y vocación durante mi tiempo como estudiante del TEC. Especialmente a los profesores Alfonso Chacón Rodríguez y Renato Rímolo Donadio que me dieron la oportunidad de participar en el desarrollo del proyecto como asistente de investigación

Por último quiero agradecer especialmente al colega, compañero y amigo Felipe Herrero Chavarría, quien fue con el que he trabajado y me ha acompañado durante los últimos 2 años tanto en cursos de licenciatura como en el desarrollo del proyecto, quien fue parte de mi formación como persona y profesional.

Dar las gracias no es suficiente para todo lo que las personas mencionadas hicieron por mí, y eternamente estaré agradecido por eso.

Bernardo Emilio Rodríguez Hall

Cartago, 26 de noviembre de 2018

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1 Introducción

1.1. Entorno e importancia del proyecto

En el del Instituto Tecnológico de Costa Rica, la Escuela de Ingeniería en Electrónica dispone de varios laboratorios enfocados a muchas áreas que posee la electrónica. El objetivo de estos laboratorios es desarrollar proyectos que aporten tanto al desarrollo de la institución como al desarrollo del país, además de contribuir con conocimiento para la comunidad científica. El DCILab, como sus siglas lo indican es el Laboratorio de Diseño de Circuitos Integrados, desarrolla proyectos de investigación en los cuales participan tanto profesores como estudiantes.

Uno de los proyectos en los que trabaja el laboratorio, está enfocado en la construcción de chips para implantes médicos. Se entiende por implantes médicos según la Administración de Drogas y Comida de Estados Unidos (FDA) por un instrumento, aparato, máquina, implemento, reactivo in vitro, que es utilizado para el diagnóstico de una enfermedad, u otra condición; o en la cura, mitigación, tratamiento, o prevención de enfermedades [9]. Esta iniciativa del laboratorio pretende junto con la Universidad Católica de Uruguay, atacar la poca disponibilidad de dispositivos médicos implantables con procesamiento avanzado, debido a que las tecnologías sobre este tipo de implantes son sensibles, complejas y están actualmente empezando su desarrollo.

Existen dispositivos médicos implantables como desfibriladores automáticos, ventiladores pulmonares, válvulas cardíacas, marcapasos, entre otros [1], pero a pesar de los grandes beneficios en la salud que estos dispositivos realizan, siempre es requerido una cirugía. Así como su nombre lo indica, para que cumplan su tarea sobre el cuerpo humano es requerido implantarlos en el tejido, implicando diseñarlos para las condiciones bajo las que estos se someten.

Actualmente el proyecto está orientado a la fabricación de una microarquitectura ISA RISC-V 32 E. Para ello, el DCILab parte del conocimiento adquirido en haber ya desarrollado una prueba de concepto de microprocesador de aplicación específica basado en esta misma plataforma RISC-V, como base para un sistema de reconocimiento de patrones orientado a la detección de disparos de armas de fuego en zonas protegidas [5, 19]. A partir de dicha experiencia, es precisamente que se propuso el proyecto de investigación bajo el cuál se enmarca este proyecto de investigación, en conjunto con la Universidad Católica de Uruguay, con financiamiento de la Agencia Nacional de Investigación e Innovación de Uruguay. El desarrollo de un chip como este es de vital importancia, ya que es el centro que toma las decisiones y las acciones que se deben realizar tanto para la adquisición de datos en los tejidos como la excitación de los mismos, además debe ser capaz de manejar las transacciones de intercambio de información con dispositivos externos que lo soliciten, como por ejemplo pruebas o incluso recolección de datos.

Entre las características más críticas para un dispositivo médico implantable está el consumo de energía, debido a que si este consume mucha energía puede provocar sobrecalentamiento y cuando se trata del cuerpo humano, se desea producir sistemas que sean compatibles con el cuerpo. Por esto la temperatura de estos dispositivos juega un papel de gran importancia para evitar que el organismo rechace el dispositivo. También, como consecuencia de un alto consumo, ocasionaría una duración insuficiente de la batería, esta situación no solo es molesta, sino que puede llegar a provocar la muerte si un dispositivo médico se queda sin alimentación antes de la duración prevista.

Varias de las ventajas de lograr un bajo consumo implicaría para los pacientes que requieren de un cambio periódico de batería, un enorme beneficio si los implantes tienen una vida útil más prolongada, puesto que la intervención quirúrgica para cambiar la batería es costosa y arriesgada. El tamaño del circuito integrado representa otro punto importante y entre menor sea el espacio que abarque menor las afectaciones para la vida cotidiana del paciente que los utilice.

1.1.1. Microprocesador RISC-V 32 E

Como es sabido, un microprocesador es el encargado de ejecutar programas basados en instrucciones programadas en lenguajes de muy bajo nivel, el cual realiza operaciones aritméticas como sumar, restar, multiplicar, dividir, operaciones lógicas binarias y accesos a memoria [13]. La micro arquitectura fundamental del chip que se diseña en el DCILab se muestra en la figura 1.1.

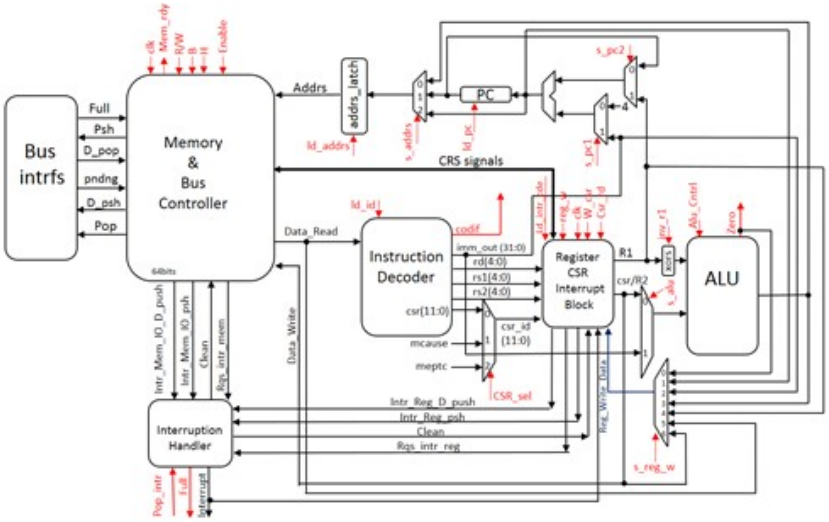


Figura 1.1: Diagrama de bloques del microprocesador RISC-V de 32 bits.

El proyecto del presente informe se enfoca y le daré especial atención en el desarrollo de sub-bloques que forman parte del MMU (Unidad de manejo de memoria) de este chip, específicamente al diseño de sub-bloques necesarios para el funcionamiento adecuado de la

memoria. Esta memoria es de tipo SRAM y fue creada por estudiantes del DCILab y aún se encuentra en etapa de desarrollo. Es por ello que la misma es actualmente comparable en complejidad y tamaño con los registros de programa del microprocesador, al poseer únicamente el arreglo de celdas conformada de 64 palabras de 32 bits cada una, para un total de 2048 bits. La verificación correcta de esta unidad inicial, abrirá la puerta al desarrollo futuro de bloques de SRAM de tamaño superior y evitar así el pagar caras licencias de propiedad intelectual (IP por sus siglas en inglés de Intellectual Property), que en el caso actual significan US$4000 que a debido desembolsar el socio del proyecto para fabricar la primera corrida del chip.

La unidad de manejo de memoria tiene 2 bloques principales que lo conforman, el controlador de bus y memoria y su respectiva memoria, como se muestran individualmente en la figura 1.2.

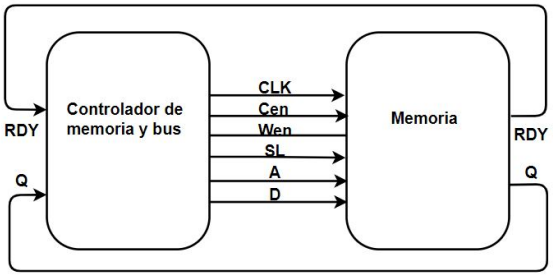


Figura 1.2: Diagrama de bloques de MMU y memoria SRAM.

La memoria de manera general está conformada por un módulo de control, un módulo de fila, un módulo de columna y la matriz o arreglo de memoria, estructura presentada en la figura 1.3. Actualmente se tiene únicamente el arreglo de memoria mencionado de 64 palabras de 32 bits en su vista del diseño físico de trazado a nivel de capas (*layout*) del arreglo de memoria.

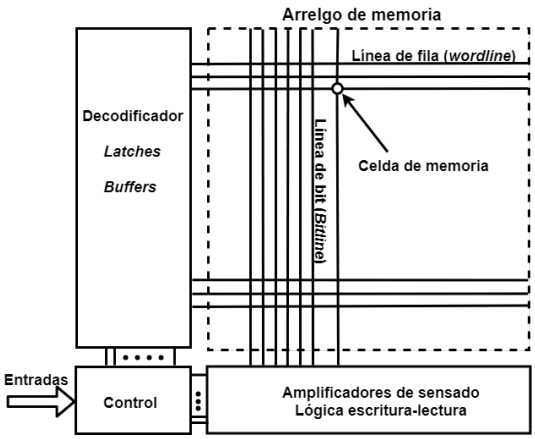


Figura 1.3: Diagrama de bloques de memoria SRAM.[3]

El proyecto descrito en este documento se centra en el desarrollo físico por medio del método *fullcustom* en su vista *layout* del decodificador de filas, necesario para poder acceder a los bits del arreglo de celdas de memoria. En la siguiente sección se hablan de los objetivos y los alcances de este proyecto.

1.2. Objetivos y alcances del proyecto

El objetivo general del presente proyecto es desarrollar el decodificador completo de filas para una memoria SRAM con capacidad de 2048 bits de almacenamiento, diseñada previamente por el autor en su trabajo como asistente de investigación en el DCILab junto con Felipe Herrero, bajo la supervisión de los ingenieros Alfonso Chacón Rodríguez, Ronny García Ramírez y Renato Rímolo Donadío. Este decodificador debe ser capaz de manejar 32 filas de 64 bits, con su respectiva señal de *enable*. Dicho decodificador deberá acomodarse dentro de un área restringida por la geometría del arreglo de celdas de memoria ya diseñado, y cumplir con una serie de restricciones de temporizado y consumo de potencia que luego se explicitarán.

Primeramente se necesita escoger una topología para el decodificador, ya que este es el sub-bloque con mayor complejidad y por ende área. Este es el sub-bloque más importante entre los circuitos de fila, por lo que debe ser dimensionado tanto para un adecuado manejo de carga, como la que este supone para los demás circuitos. Se harán varias simulaciones de los caminos críticos de este decodificador para su posterior optimización y dimensionamiento. Seguidamente, debe ser integrado el sub-bloque del *enable* que permite el paso de las señales de salida del decodificador hacia las líneas de fila de la memoria, los cuales deben ajustarse al trazado de la memoria. Finalmente diseñar e interconectar con los bits de direcciones del decodificador, los *latches* que sostienen las direcciones de entrada para un correcto acceso a los espacios de memoria. Se elegirá la opción que suponga menos transistores y por ende menos área.

El desarrollo individual de los 3 bloques anteriores y su interconexión se encuentran bajo una serie de limitaciones y debe de cumplir con las siguientes especificaciones:

1. Todos los circuitos diseñados deben ser alimentados y funcionales con una alimentación de 1.8 V.
2. Los diseños del decodificador de fila deben realizarse en la herramienta de Synopsys Custom Compiler.
3. Los diseños de los circuitos de fila deben ser diseñados en la tecnología CMOS de 180 nm y fabricables.
4. Diseñar a nivel de *layout* el circuito de los *latches*. Las simulaciones con parásitas extraídas debe cumplir un t_{pd} menor a los 3 ns.
5. Diseñar a nivel de *layout* el circuito de los *buffers* que manejen la carga de las línea de fila de la memoria con un tiempo de propagación menor a los 2 ns.
6. Diseñar a nivel de *layout* en su modelo extraído, el circuito del decodificador con un tiempo de propagación menor a los 5 ns.
7. Integrar todos los módulos del decodificador de fila con la memoria, para una frecuencia objetivo de reloj de al menos 20MHz.
8. Dimensionar el área de interconexión tal que ocupe una altura máxima de 300μm, para ajustarse con el arreglo de celdas de memoria ya diseñado.

2 Marco teórico

En esta sección se verán conceptos de que es un transistor basado en la tecnología MOS y su funcionamiento, y se estudiarán sus características más importantes para la construcción de circuitos digitales. También se incluyen conceptos de lógica combinacional, como álgebra booleana, bits LSB y MSB, tablas de verdad y mintérminos. Otros temas que se mencionarán son el funcionamiento básico de memorias SRAM y el papel que cumple su decodificador, los sub-bloques que son parte de este decodificador como sus *buffers* de salida y los *latches* de entrada. Finalmente se hablará de los modelos de retardo y la teoría detrás del dimensionamiento de compuertas basado en el esfuerzo de camino y su optimización.

2.1. Transistores en tecnología MOS

El Silicio (Si) es el semiconductor base de la mayoría de los circuitos integrados[22]. El Silicio consiste en una red de átomos tridimensional del grupo químico numero IV, lo que posibilita formar enlaces covalentes con 4 átomos adyacentes. La conductividad del silicio puede modificarse si se introducen impurezas llamadas dopantes en su red cristalina, como se muestran el la figura 2.1.

Por ejemplo un dopante del grupo químico número V como el Arsénico, con 5 electrones de valencia, puede reemplazar un átomo de Silicio en el enlace. El electrón sobrante puede desprenderse del átomo de As simplemente por vibración térmica. Este quinto electrón puede luego usarse para generar una corriente eléctrica (se aumenta la conductividad del semiconductor por ende). A este tipo de semiconductores dopados se les llama tipo N, al generar un exceso de portadores con carga negativa.

De la misma manera un dopante del grupo químico III como el Boro, con 3 electrones de valencia, deja un enlace descubierto en la red cristalina (un "hueco" en la jerga de semiconductores). Este enlace puede ocuparse por vibración térmica por un electrón proveniente de alguno de los átomos de Si circundantes. Este hueco actúa como carga positiva y a los materiales dopados de esta manera se les llama dopante tipo P.

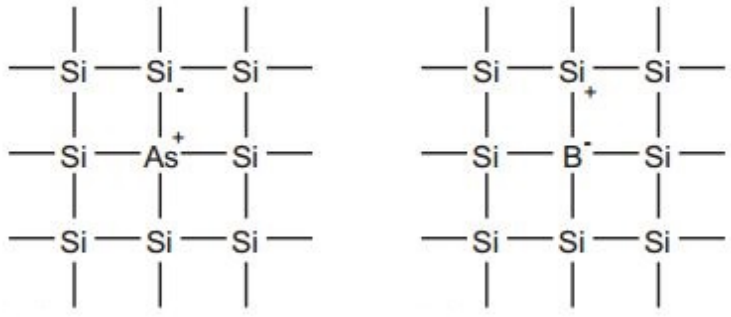


Figura 2.1: Red cristalina del Silicio con dopantes N y P.[7]

Una unión entre semiconductores tipo N y P es llamado diodo. Cuando el tensión en el semiconductor tipo P es mayor sobre el tipo N, se dice que el diodo está polarizado en directa y este permite el paso de corriente, y si el voltaje es igual o menor que en el semiconductor tipo P sobre el tipo N, se dice que el diodo está polarizado en reversa y fluye muy poca corriente a través de este.

Una estructura Metal-Óxido-Semiconductor (MOS) es creada superponiendo varias capas de materiales conductores y aislantes. Estas estructuras son fabricadas sobre redes muy puras de Silicio llamadas obleas y usando una serie de procesos químicos como la oxidación del Silicio, introducción de dopantes, deposición y colocación de cables y contactos metálicos. En la tecnología CMOS (Complementary Metal-Oxide-Semiconductor) existen 2 tipos de transistor, basado en dopantes tipo N (NMOS) y basado en dopantes tipo P (PMOS) y como el funcionamiento de estos transistores está fundamentado en el control por campos eléctricos, también a estos dispositivos se les llama transistores de efecto de campo Metal-Óxido-Semiconductor (MOSFET). A continuación en la figura 2.2 se muestran los transistores NMOS y PMOS desde una perspectiva transversal.

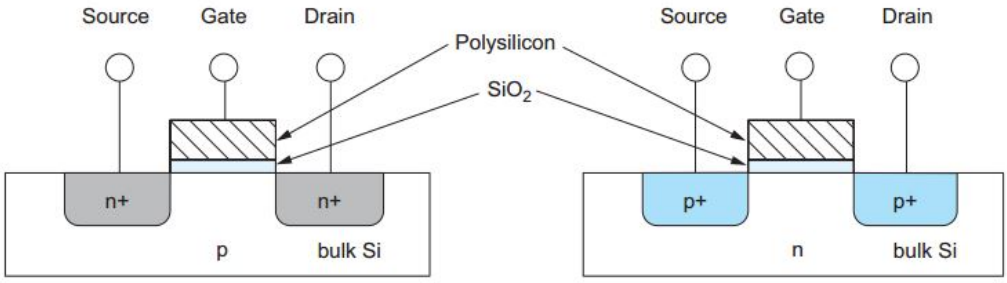


Figura 2.2: Transistores NMOS y PMOS.[7]

En la figura 2.2 las regiones n+ y p+, indican que la concentración del dopante es mayor para semiconductor tipo N y tipo P respectivamente. Cada transistor consiste de una compuerta de algún material conductor, una capa aislante generalmente de dióxido de silicio (SiO2)[7] y la oblea de silicio también llamado substrato o cuerpo.

Un transistor NMOS está construido sobre un substrato tipo P y posee regiones de semiconductor tipo N adyacentes a la compuerta llamados drenador y surtidor o fuente, estas regiones son físicamente equivalentes. Al igual que el transistor NMOS, el PMOS posee la misma estructura con la diferencia de que el substrato es de tipo N y las regiones del drenador y surtidor son de tipo P.

La compuerta o gate es la terminal del transistor por medio del cual se controla el flujo eléctrico entre las terminales del drenador y surtidor. En el caso del NMOS la terminal del substrato está generalmente conectada a tierra por lo que las uniones n-p entre el surtidor y drenador y substrato están polarizados en reversa, además si la compuerta también está conectada a tierra no circulará corriente a través del transistor. En este caso, el transistor está apagado. Si la tensión en la compuerta aumenta, se crea un campo eléctrico que atrae cargas negativas debajo del aislante y si el voltaje es lo suficientemente alto la cantidad de electrones sobrepasa la de huecos, formando una región llamada canal, que actúa como semiconductor tipo n, de esta forma es posible conducir corriente eléctrica entre drenador y surtidor. En esta situación se dice que el transistor está encendido.

Para un transistor tipo P la situación es contraria, el substrato está conectado a un voltaje positivo y cuando la compuerta está conectada a un voltaje positivo las uniones p-n están polarizadas en reversa, la corriente no fluye y el transistor está apagado. Cuando la tensión de la compuerta disminuye, las cargas positivas son atraídas debajo de la compuerta, y si este voltaje es lo suficientemente bajo se forma un canal o camino conductor, sobre el cual puede fluir corriente entre surtidor y drenador y el transistor está encendido.

El voltaje positivo es llamado usualmente V_{DD} y representa un 1 lógico en circuitos digitales, para este proyecto el valor de VDD es igual a 1.8 V. El voltaje bajo es llamado VSS o ground (GND). En resumen un transistor MOS controla el paso de la corriente entre surtidor y drenador, si se simplifica mucho este modelo, los transistores pueden ser vistos como un "switch". Cuando en la compuerta de un NMOS hay un 1 el transistor se enciende, y cuando hay un 0 el transistor se apaga y en un PMOS es lo opuesto, con 1 en la compuerta el transistor se encuentra apagado y con un 0 encendido, como se muestra en la figura 2.3

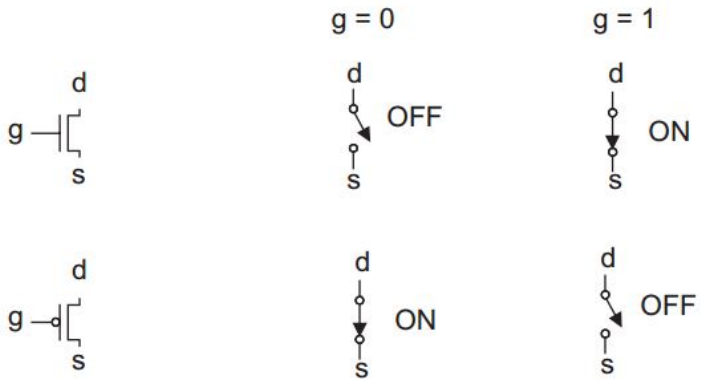


Figura 2.3: Símbolo de transistores y sus modelos como interruptor.[7]

2.1.1. Modelo de capacitancias parásitas del transistor MOS

El transistor construido usando MOS posee capacitancia en cada una de las terminales debido a su construcción. Estas capacitancias por lo general son no lineales y dependen del voltaje (C-V). A pesar de esto es posible aproximar como simples capacitores cuando su comportamiento es promediado en las transiciones de voltaje de una compuerta lógica. Esta sección presenta fundamentos del modelado de estas capacitancias, para futuras estimaciones en los retrasos de las señales.

La compuerta de un transistor MOS es estructuralmente como un capacitor de placas paralelas y es necesario que funcione como tal, ya que se requiere carga para obtener más altas corrientes de saturación. A continuación, se presenta la ecuación para obtener la capacitancia de un capacitor de placas paralelas con una delgada capa de dieléctrico de óxido entre ellas [7]:

$$C_g = C_{OX} W L \tag{1}$$

Suponiendo que el transistor no está saturado, el canal se forma desde el drenador hasta el surtidor y actúa como una de las placas del capacitor. Usualmente se aproxima esta capacitancia entre la terminal de la compuerta y el substrato, y se le dice C_{gs}.

La mayoría de los transistores usados en lógica digital son del largo mínimo debido a la alta velocidad de conmutación y al bajo consumo dinámico de potencia [7]. Por esto se normaliza la capacitancia sobre este largo (L) característico de la tecnología y con este se establecerá la base sobre la cual, solo será necesario saber el ancho (W) para obtener la capacitancia de entrada del transistor. Por medio de este largo característico se define lo siguiente:

$$C_g = C_{permicron} \times W; C_{permicron} = C_{OX} L = \frac{\epsilon_{ox}}{t_{ox}} L \tag{2}$$

Además de la capacitancia de la compuerta, las terminales del drenador y el surtidor también poseen una capacitancia asociada que impacta el desempeño del circuito. Estas capacitancias provienen de la uniones p-n entre las difusiones de estas terminales y el substrato, por esto se les llama capacitancias de difusión C_{sb} y C_{db}. Esta capacitancia se origina de la región de vaciamiento, la cual es una zona donde no hay portadores de carga alrededor de la junta y que actúa como aislante entre las regiones conductoras de tipo P y N. Asimismo la capacitancia es dependiente del área y del perímetro de las difusiones, la profundidad de la difusión, los niveles de dopado y la tensión. Como las difusiones poseen una alta capacitancia y resistencia , generalmente se construyen los más pequeñas posible en el trazado. En la Figura 2.4 se muestran tres posibles conexiones para dos transistores en serie.

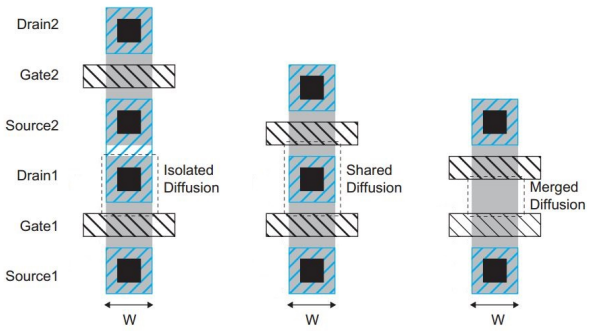


Figura 2.4: Prelayout de las geometrías de los diferentes tipos de difusión.[7]

Los primeros transistores en serie de la izquierda poseen su contacto a difusión en cada surtidor y drenador. El la segunda configuración, los transistores en serie del medio tienen el drenador y el surtidor compartiendo un solo contacto a difusión y en el último par de transistores en serie se elimina el contacto y se mezclan las difusiones en una sola región. El primer modelo sería el más lento, por que tiene 2 contactos que poseen capacitancias que deben ser cargadas, luego el segundo modelo que comparte un contacto y el más veloz sería el que no posee los contactos, por lo que es deseable evitarlos cuando es posible.

Para modelos no muy complejos, se puede aproximar el valor de las capacitancias de difusión, semejantes al de la de compuerta [7].

Además de las capacitancias anteriormente mencionadas, existen otras llamadas capacitancias de traslape que provienen de como su nombre lo indica, es una sobreposición o traslape de la compuerta sobre el drenador y el surtidor, como se muestran en la figura 2.5.

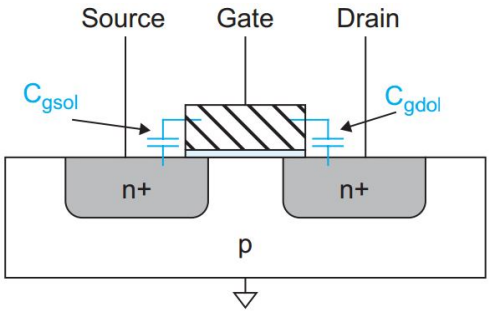


Figura 2.5: Capacitancias de traslape del transistor MOS.[7]

Estas capacitancias son proporcionales al ancho del transistor y deben ser agregadas a la capacitancia intrínseca de la compuerta, ya que es conveniente tomar la capacitancia de la compuerta como un solo capacitor conectado a esta terminal para una mejor aproximación.

2.2. Memorias SRAM

En esta sección se hablará sobre las diferentes estructuras básica de memorias SRAM, con especial énfasis en arreglos de memoria compuestos por celdas de 6 transistores (6T), los bloques que la conforman y su principio de funcionamiento. Además de exponer los circuitos y funciones de los sub-bloques del decodificador de fila que forman parte de esta memoria.

2.2.1. Arquitectura de SRAM

Las celdas de Static Random Access Memory (SRAM), como su nombre lo indica, no requieren constante refrescamiento de datos para evitar una posible pérdida de información y esta información no se perderá mientras se mantenga alimentada la memoria. Esta es una de las mayores ventajas que poseen las SRAM sobre DRAM [4].

Lo que permite preservar los datos estáticos en las celdas SRAM es la realimentación entre los circuitos, a diferencia de las DRAM que están compuesta únicamente de un capacitor donde se almacena la carga y un transistor de acceso a este capacitor. En consecuencia de las corrientes de fuga a través del transistor es que es necesario constantemente leer y reescribir los datos bit por bit. Las SRAM son más rápidas, pero requieren una mayor cantidad de área por bit que las DRAM, aun así la densidad de memoria que se puede integrar sigue siendo mayor que relizar arreglos con flip-flops [7].

Las celdas de memoria pueden tener uno o más puertos de acceso. En el caso de una memoria de escritura-lectura, un puerto puede ser de escritura solamente, lectura solamente o capaz de ambos procesos.

Un arreglo de memoria contiene 2^n palabras o *wordlines* de 2^m bits cada una, donde cada bit es almacenado en una celda. La figura 2.6 muestra la organización de un arreglo de memoria que contiene 16 palabras de 4 bits cada una, este ordenamiento forma una especie de torre rectangular. Cuando se realizan accesos, el decodificador de fila utiliza la dirección para activar la *wordline* especificada, luego las celdas de esta *wordline* se conectan a las lineas de bit, las cuales son condicionadas a un valor en particular antes de los accesos. Los circuitos de columna pueden tener amplificadores para agilizar el proceso de lectura o solamente *buffers* para extraer los datos de la memoria.

Para casos en que los arreglos son muy grandes, como en escalas de miles de palabras en adelante, no se mantiene la estructura rectangular mencionada, ya que la altura de este bloque sería enorme y colocar este arreglo en el espacio reducido de la oblea con los demás circuitos y los cables que estos implican, imposibilitarían su fabricación. Sin embargo existen arreglos que desdoblan las *wordlines* para generar más columnas con lo que las filas de la memoria son ahora de 2^k palabras, pero el arreglo físicamente está organizado como 2^{n-k} filas de 2^{m+k} bits. También en la figura 2.6 se muestra un arreglo con $k = 1$ con 8 filas y 8 columnas, debido a esto es necesario un decodificador de columna que controle un

multiplexor para seleccionar 2^m bits de la *wordline* escogida para llegar a la información deseada. Generalmente en memorias de grandes escalas se construyen múltiples pequeños sub-arreglos para que las palabras y bits se conserven lo suficientemente cortas y rápidas.

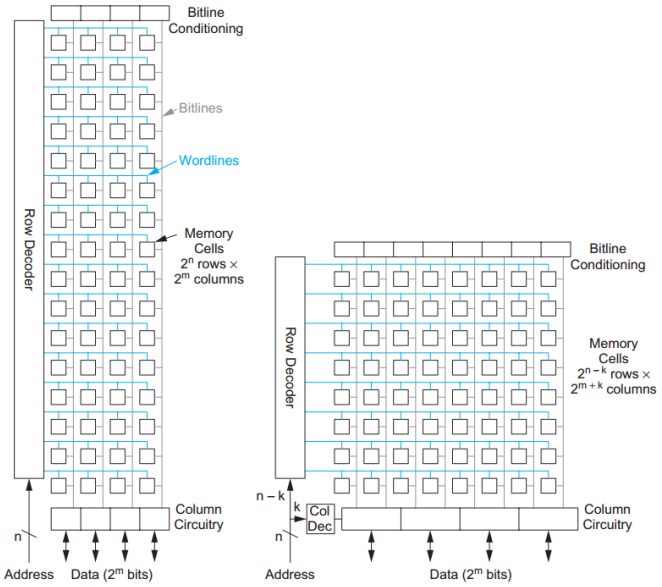


Figura 2.6: Estructuras de arreglos de memoria.[7]

La estructura del arreglo de memoria SRAM para la cual se está diseñando el decodificador, entra en el ordenamiento en donde las *wordlines* no poseen más de un espacio de memoria por cada una, como el primer tipo de estructura mencionado.

2.2.2. Celda SRAM de 6T

Las celdas de SRAM deben ser capaces de escribir y leer información, así como deben mantenerla mientras la alimentación este conectada. Estas funciones son lógrales con flip-flops pero a un costo que conlleva la cantidad de transistores y el área que estos abarcan. Las celdas de 6T cumplen con los requisitos mencionados a un costo de mayor lógica para los circuitos periféricos que las acompañan.

La estructura convencional de una memoria SRAM de 6 transistores usa 2 inversores realimentados y por medio de 2 transistores NMOS de acceso se proporciona una conexión de la celda con el mundo exterior. En la figura 2.7 se muestra la configuración del circuito para esta celda de 6T.

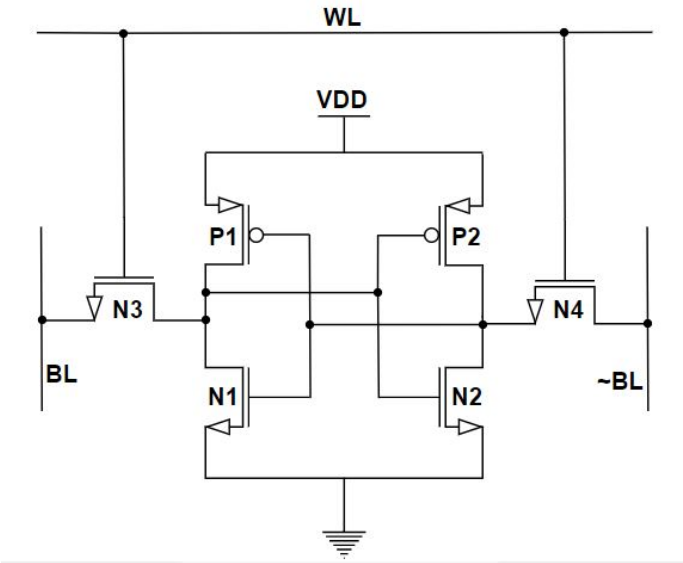


Figura 2.7: Circuito de celda SRAM de 6T.

Para esta celda existen 3 modos de operación: reposo, lectura y escritura. Durante el modo de reposo la señal de *wordline* (WL) de la figura 2.7 posee un valor de 0, apagando los transistores N3 y N4 de acceso. Esto evita el acceso a los datos del par de inversores que permanecen estáticos, esto retiene su estado mientras se mantenga la alimentación. Antes de cualquier escritura o lectura siempre se condicionan las lineas de bit a valores como V_{DD} o V_{DD} medios, ya que esto acelera las transiciones en estos nodos cuando se realiza alguna transacción.

Durante el modo de operación de lectura se da acceso a los bits que contienen los datos con la señal WL en 1, esto enciende los transistores de acceso para poder transmitir los datos de los inversores hacia las líneas de bit. Para que sea posible transmitir los datos correctamente a las lineas de bit sin que sobre escriba la memoria, es necesario respetar un dimensionamiento en el cual los transistores N1 y N2 sean más fuertes que los transistores de acceso N3 y N4, ya deben ser capaz de sostener su valor valor cercano a tierra con tal de evitar que se supere el punto de inversión y consecuente se ejecute una escritura no deseada.

De manera similar durante una escritura la señal de WL es igual a 1, pero esta vez los datos a almacenar se cargan en las líneas de bit para escribir en los inversores. A pesar de que es sencillo establecer los valor en las lineas de bit, por la mismas razones que hacen que en una lectura los datos no se dañen, es obligatorio presionar los valores de las lineas de bit sobre los transistores P1 y P2 a través de los transistores N1 y N2 de acceso, esto impone otra regla de dimensionamiento en donde los transistores de acceso deben ser más fuertes que los transistores PMOS de los inversores, con el fin de ejecutar las escrituras correctamente.

2.3. Circuitos de fila de SRAM

Los circuitos de fila que forman parte de los sub-bloques internos de la memoria son los *latches* o cerrojos, el decodificador y los *buffers* o manejadores de línea. Cada uno de estos bloques realiza una tarea en específico que permite un correcto funcionamiento cuando la memoria se encuentra en cualquiera de sus 3 modos de operación.

A continuación en la figura 2.8 se muestra la estructura de como los sub-bloques anteriores se interconectan entre sí y la memoria.

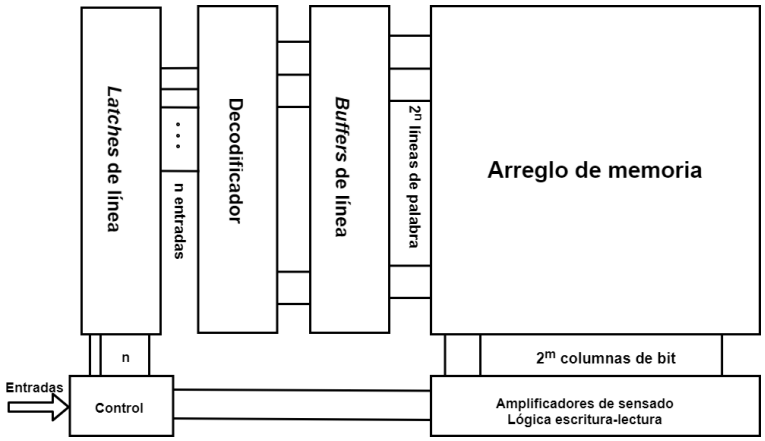


Figura 2.8: Diagrama de bloques de circuitos de fila.[3]

Las direcciones son pasadas desde el sub-bloque de control hacia los cerrojos del decodificador, una vez la señal de reloj lo permita, estas direcciones son capturadas, esto impide que otras señales no deseadas sobre las direcciones afecten sus valores e incurra en algun error posterior, luego esta dirección es decodificada y propagada hacia los circuitos de habilitación y los manejadores de fila, una vez las señales de salida del decodificador llegan al circuito del *enable* o habilitador, se espera a la señal de habilitación para dar paso hacia el último sub-bloque que es capaz de manejar toda la carga que supone las palabras de la memoria.

2.3.1. Cerrojos del decodificador

Los *latches* o cerrojos son elementos secuenciales básicos usados en el diseño de circuitos integrados usados para aplicaciones de almacenamiento de datos. Se construyen usando compuertas de paso o *buffers* triestado para transferir los datos mientras que el cerrojo es transparente y para realimentar los datos capturados mientras el cerrojo está en retención. Un ejemplo de la estructura anterior descrita se muestra en la figura 2.9.

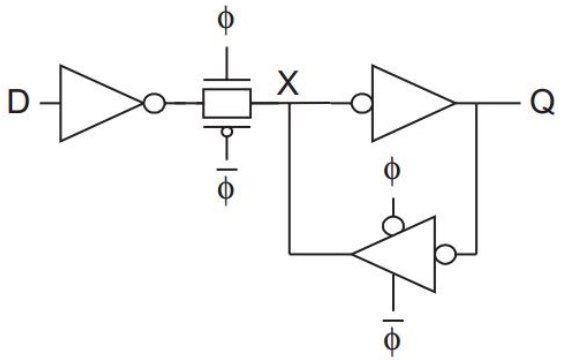


Figura 2.9: Diagrama de ejemplo de un cerrojo típico.[7]

A nivel de circuito este funciona de modo que mientras la señal ϕ es 1, esto enciende ambos transistores de la compuerta de paso, lo que permite transferir los datos del primer al segundo inversor y seguidamente a la salida, todo esto entretanto el inversor triestado del lazo de realimentación está en alta impedancia. En el tiempo que este periodo se dice que el cerrojo es transparente. En el caso de que la señal ϕ es 0, se apagan los transistores de la compuerta de paso impidiendo el paso de los datos, mientras que el dato anterior se retiene y se realimenta por medio del inversor triestado. De esta forma se almacenan los datos y se dice que el cerrojo está en retención.

Para este tipo de circuitos en el que se deben capturar datos, es necesario conocer el tiempo de *setup*, este es el tiempo mínimo necesario con el que el dato a almacenar debe estar listo o estable en la entrada **D** del cerrojo. Si no se cumple este tiempo el dato no será capturado con éxito[8].

2.3.2. Decodificador de fila

El decodificador es un de los bloques funcionales más importantes en la electrónica digital, este se encuentran en circuitos de selección o direccionamiento e incluso puede ser utilizados para funciones lógicas. Específicamente los decodificadores de direccionamiento juegan un papel muy destacado en el diseño de memorias y, debido a la gran cantidad de celdas de almacenamiento, existen muchos diseños orientados a la reducción del consumo de potencia y al incremento del desempeño.

El decodificador de fila usa la dirección capturada por los cerrojos en la entrada para activar una de las *wordlines* durante una escritura o lectura, de modo que solo se pueda acceder a un solo espacio de memoria por acceso. La estructura más simple del decodificador está basada en compuertas AND usando los bits de dirección y su complemento, como se muestra de ejemplo en la figura 2.10.

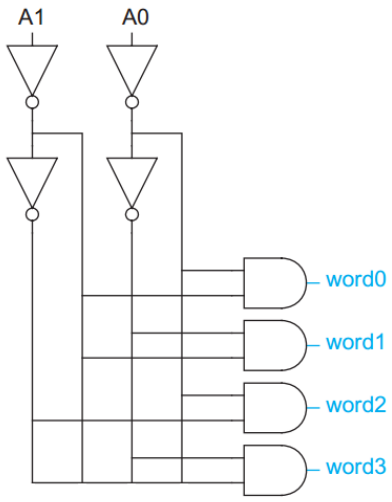


Figura 2.10: Decodificador de 2 por 4 bits con compuertas AND.[7]

Los decodificadores típicamente tienen un alto esfuerzo eléctrico y de bifurcación, por lo que se usa la predecodificación en varias etapas para minimizar el esfuerzo lógico con árboles de NANDS y NORS de 2 ó 3 entradas junto con inversores para reducir el esfuerzo lógico y aumentar la velocidad. La predecodificación consiste en que al existir varias compuertas que comparten exactamente las mismas entradas, estas se pueden factorizar para mejorar el diseño del decodificador. Esta técnica no mejora el esfuerzo de camino del decodificador pero mejora el área y la distribución de los transistores en el *layout* [7].

2.3.3. Manejadores de línea

Se sabe que en los circuitos integrados modernos, el retraso de las interconexiones contribuyen en gran parte al total del retraso, dado que el retaso incrementa críticamente con la longitud del cable. Los *buffers* se usan tradicionalmente para linealizar la dependencia del retraso de las señales con la longitud de la interconexión. Además es importante mencionar que colocar *buffers* posee sus desventajas como el incremento del área, ya que agregar circuitos siempre requiere espacio e incluso reorganización de los *layouts* más el consumo adicional que estos implican [10].

A pesar de las ventajas y desventajas es necesario colocar estos circuitos para así poder manejar las *wordlines* de las memorias, ya que son muy capacitivas debido a la cantidad de transistores que se conectan a estas lineas. En consecuencia es requerido dimensionarlos para lograr mover estas cargas adecuadamente.

2.4. Modelos de retardo

Para esta sección es necesario establecer algunas definiciones generales para poder tener una métrica consistente de los retrasos y transiciones que pueden sufrir las señales.

- Tiempo de propagación (t_{pd}): Máximo tiempo desde que la señal de entrada está al 50 % de su transición hasta que la señal de salida este a su 50 % de transición, siendo este valor igual a $\frac{VDD}{2}$.
- Tiempo de contaminación (t_{cd}): Mínimo tiempo desde que la señal de entrada está al 50 % de su transición hasta que la señal de salida este a su 50 % de transición, siendo este valor igual a $\frac{VDD}{2}$.
- Tiempo de subida (t_r): Tiempo necesario para que una señal suba del 20 % al 80 % del valor de alimentación.
- Tiempo de bajada (t_f): Tiempo necesario para que una señal baje del 80 % al 20 % del valor de alimentación.

Se sabe que cuando una señal de entrada cambia, la salida retendrá su valor anterior al menos t_{cd} y lo máximo será t_{pd}. En ocasiones para diferenciar entre los retrasos para pendientes de subida se nombran t_{pdr} y t_{cdr} para tiempos de propagación y contaminación respectivamente, de las misma manera que para las pendientes de bajada t_{pdf} y t_{cdf}. A las compuertas que cargan y descargan nodos se les llama *drivers* y a las compuertas que deben ser manejadas por el *driver* se les llama carga.

2.4.1. Modelo de retardo RC

El modelo de retardo RC aproxima las características no lineales de corriente-tensión y capacitancia-tensión del transistor con una resistencia y capacitancia promedio sobre las transiciones de operación de las compuertas. Por medio de este modelo es posible aproximar y realizar estimaciones sobre los retardos.

Este modelo trata al transistor como un interruptor en serie con una resistencia, esta resistencia efectiva es el promedio de $\frac{V_{ds}}{I_{ds}}$ en el intervalo de transición de interés. Se define el transistor NMOS unitario, el cual tiene tiene una resistencia efectiva de R. El tamaño del transistor unitario es arbitrario, pero por convención se refiere a un transistor de dimensiones 4/2 λ, (W/L) donde lambda es la mitad del largo característico de la tecnología. Las herramientas permiten un tamaño mínimo de transistor de 220 nm/180 nm (2.44/2 λ).

Basado en lo anterior, si un transistor NMOS es k veces el ancho unitario se dice que este tiene resistencia igual a R/k, por que este entrega k veces más corriente, y en el caso del

transistor unitario PMOS, ya que posee una mayor resistencia comparado con el transistor NMOS, generalmente en un rango de 2R a 3R por la movilidad, lo que supone una relación de tamaños P a N de 2 a 1 [7].

Se define C igual a la capacitancia de compuerta de un transistor unitario, por lo que un transistor de k veces el ancho mínimo tiene una capacitancia igual kC. La capacitancia de difusión depende de los tamaños de las regiones del drenador y surtidor, pero como ya antes se mencionó estas regiones también poseen un valor cercano a C. Los transistores más anchos tienen capacitancia proporcionalmente más grande en las difusiones, pero incrementar la longitud solo aumenta la capacitancia de compuerta pero no afecta la capacitancia de difusiones.

Con las aproximaciones anteriores se puede llegar a modelar los transistores NMOS y PMOS como se muestran a continuación en la figura 2.11:

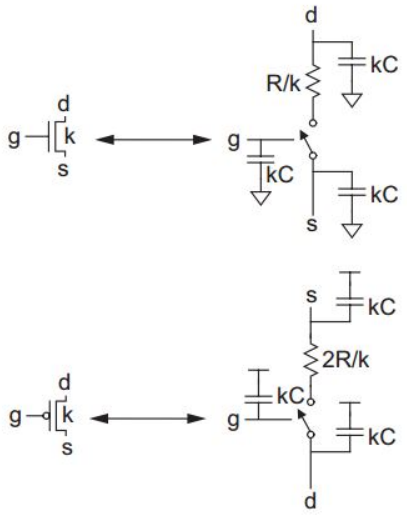


Figura 2.11: Circuitos equivalentes de transistores NMOS y PMOS .[7]

En la figura 2.12 se muestra un ejemplo de 2 inversores conectados en cascada, su equivalente RC y la red final RC.

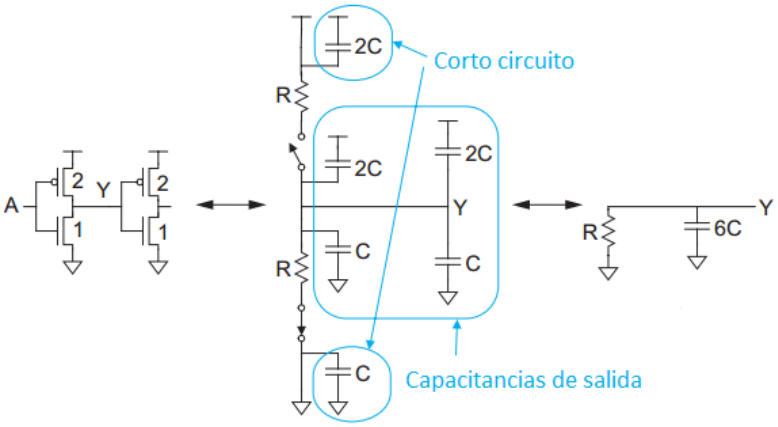


Figura 2.12: Equivalente RC de inversores en cascada.[7]

Se eliminan varias capacitancias que se encuentran en corto circuito y se suman todas las capacitancias en el nodo Y de salida. Una de las características más importantes a notar, es que para todas las transiciones de las compuertas se desea un resistencia igual a la unitaria que maneje la carga, tanto para las pendientes de subida como de bajada, esta es la razón por la cual en el modelo equivalente es indiferente cual transición se realiza, mientras se respete la relación 2 a 1 de los transistores.

Ahora considérese aplicar el modelo RC para estimar la respuesta al escalón de un circuito de primer orden de la figura 2.13, el cual es un modelo de un inversor dimensionado para iguales pendientes de subida y bajada.

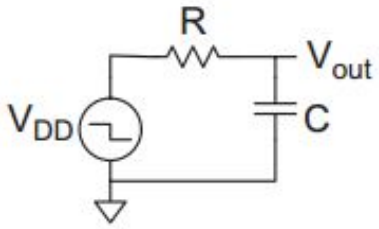


Figura 2.13: Circuito equivalente RC de inversor unitario.[7]

Este sistema tiene función de transferencia y respuesta al escalón

$$H(s) = \frac{1}{1 + sRC} \qquad V_{out}(t) = V_{DD}e^{-t/\tau}$$

Con $\tau = RC$ se tiene que si la tensión en el nodo V_{out} alcanza $V_{DD}/2$, este será el tiempo de propagación, el cual es $t_{pd} = RCln2$, según la figura 2.14.

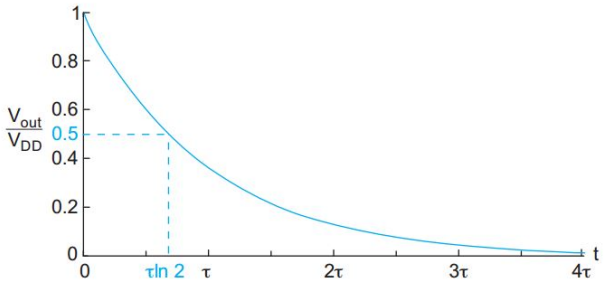


Figura 2.14: Respuesta al escalón de primer orden.[7]

El factor $ln2 = 0{,}69$ es incómodo y la resistencia efectiva R es al final una aproximación empírica, en consecuencia por simplicidad se incorpora este valor a la R efectiva $R' = Rln2$ y por conveniencia solo se denota como R.

Ahora en la figura 2.15 se muestra un sistema de segundo orden:

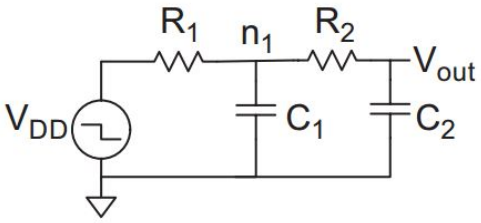


Figura 2.15: Circuito RC de segundo orden.[7]

Donde la función de transferencia y la respuesta al escalón son:

$$H(s) = \frac{1}{1 + s[R_1C_1 + (R_1 + R_2)C_2] + s^2R_1C_1R_2C_2} \quad V_{out}(t) = V_{DD}\frac{\tau_1 e^{-t/\tau_1} - \tau_2 e^{-t/\tau_2}}{\tau_1 - \tau_2}$$

Con

$$\tau_{1,2} = \frac{R_1C_1 + (R_1 + R_2)C_2}{2}\left(1 \pm \sqrt{1 - \frac{4\frac{R_2}{R_1}\frac{C_2}{C_1}}{[1 + (1 + \frac{R_2}{R_1})\frac{C_2}{C_1}]^2}}\right)$$

Como es notable las 2 ecuaciones anteriores son bastante extensas y de difícil deducción, lo que en un principio elimina el propósito de simplificar los circuitos CMOS en una red equivalente RC, sin embargo es posible aproximarla a un sistema de primer orden con una sola constante de tiempo:

$$\tau = \tau_1 + \tau_2 = R_1C_1 + (R_1 + R_2)C_2 \tag{3}$$

La aproximación de la ecuación (3) funciona mejor si una de las constantes es significativamente mayor que la otra [15]. En el caso donde $R_1 = R_2 = R$, $C_1 = C_2 = C$, se tiene que $\tau_1 = 2{,}6RC$, $\tau_2 = 0{,}4RC$ y $\tau = 3RC$, y por lo tanto la respuesta del sistema de primer orden se asemeja al de segundo orden. El error en la estimación del tiempo de propagación es menor al 7 %, incluso en el peor de los casos donde ambas constantes son igual el error es menor al 15 %, como se puede ver en la figura 2.16. Cabe mencionar que es una mala aproximación si se desea usar una sola constante para nodos intermedios como n_1 de la figura 2.15, para este proyecto es de interés solo obtener tiempos sobre el nodo de salida donde la aproximación funciona.

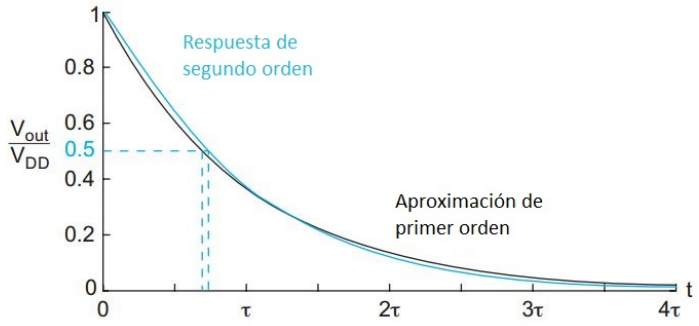


Figura 2.16: Gráfica comparativa entre la aproximación y la respuesta de segundo orden.[7]

2.4.2. Modelo de retardo de Elmore

La mayoría de los circuitos a desarrollar pueden ser descritos con árboles RC, la raíz de este árbol son las fuentes de tensión y las hojas son los capacitores al final de las ramificaciones. El modelo de retardo de Elmore calcula el retardo desde la fuente conmutando hasta el nodo de la hoja que se está cambiando, como la suma de cada una de las capacitancias C_i en los nodos i, multiplicado por la resistencia efectiva R_{is} en el camino compartido desde la fuente y el nodo y la fuente y la hoja de salida. [21].

$$t_{pd} = \sum_i R_{is}C_i \tag{4}$$

De manera general para la estimación del tiempo de propagación para un inversor unitario que tiene conectado a la salida m inversores iguales, por cada inversor de carga se tiene una capacitancia de compuerta de 3C para un total de 3mC. El nodo de salida ve una capacitancia parásita de 3C intrínseca debido a las difusiones de los drenadores. En la figura 2.17 se muestra la red RC equivalente.

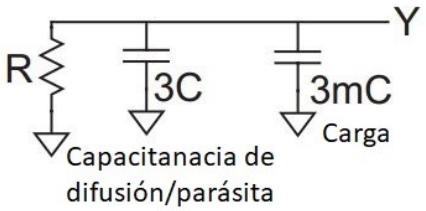


Figura 2.17: Red RC equivalente de inversor unitario con m inversores de carga.[7]

Para el circuito de la figura anterior según el modelo de Elmore el retardo sería igual a $t_{pd} = (3 + 3m)RC$, del mismo modo se establece de manera general que el primer inversor sea w veces el ancho unitario, por lo que la resistencia efectiva es R/w y las capacitancias crecen a Cw, y una vez más con el modelo de Elmore el retardo es:

$$tpd = ((3w + 3m)C)(R/w) = (3 + 3m/w)RC \tag{5}$$

Del retardo anterior se define el fanout de una compuerta h, como la relación entre la capacitancia de entrada y la de salida.

$$h = \frac{3mC}{3wC} = \frac{m}{w} \tag{6}$$

O de forma general si no se tienen copias idénticas de carga en la salida de la compuerta como:

$$h = \frac{C_{out}}{C_{in}} \tag{7}$$

También para expresar el retardo independientemente de la tecnología, se denota el retardo normalizado d relativo al retardo del inversor unitario con un mismo inversor de carga sin capacitancia parásita ($\tau = 3RC$).

$$d = \frac{t_{pd}}{\tau} \tag{8}$$

Como se puede observar en la ecuación (5), el retardo posee 2 componentes, el retardo parásito, el cual es el tiempo para que la compuerta pueda cargar o descargar su propia capacitancia de difusión intrínseca y el "esfuerzo de retardo" que depende de h.

2.4.3. Modelo lineal de retardo

El modelo RC mostró que el retardo es una función lineal del fanout de una compuerta. Basado en esta observación se simplifica el análisis caracterizando una compuerta por su pendiente y su intersección en el eje y. El retardo normalizado de una compuerta puede ser expresado en unidades de τ como:

$$d = f + p \tag{9}$$

Donde p es el retardo parásito y a f el esfuerzo de etapa y depende de la complejidad de la compuerta y el *fanout*.

$$f = gh \tag{10}$$

La complejidad de la compuerta se representa como el esfuerzo lógico g y se define como la relación entre la capacitancia de entrada de un compuerta entre la capacitancia de entrada de un inversor que puede entregar la misma corriente. Puede ser estimado por medio del circuito a nivel de transistor usando los anchos de los transistores de tal manera que se tenga resistencia unitaria, como anteriormente se mencionó. El inversor presenta 3 unidades de capacitancia, en consecuencia posee esfuerzo lógico de 1, una NAND de 3 entradas presenta una capacitancia de entrada de 5 unidades, por lo que tiene esfuerzo lógico de 5/3 y para una NOR de 3 entradas, presenta una capacitancia de entrada de 7 unidades para un esfuerzo lógico de 7/3 [7]. En la figura 2.18 se muestra las tres compuertas anteriores a nivel de transistor con el ancho respectivo, la capacitancia total en sus entradas y su esfuerzo lógico.

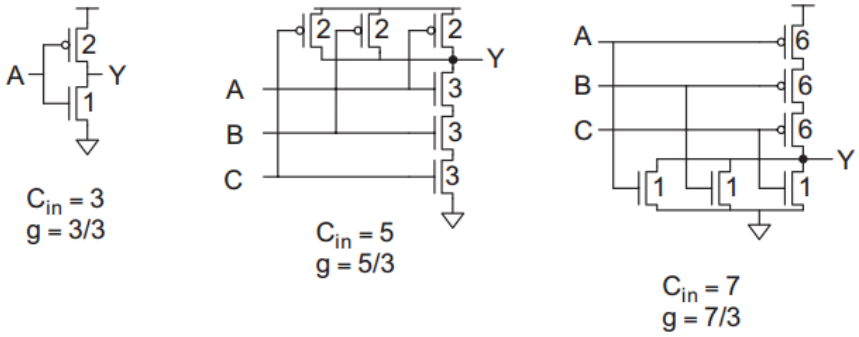


Figura 2.18: Esfuerzo lógico de varias compuertas .[7]

Como es de esperar, al agregar más entradas se requieren más transistores y de mayor ancho que manejen esas entradas, por lo que el esfuerzo lógico aumenta, en la tabla 2.1 se muestran como crece y se generaliza con una ecuación para n entradas.

Tabla 2.1: Esfuerzos lógicos de compuertas lógicas.

Compuerta	Número de entradas				
	1	2	3	4	n
Inversor	1				
NAND		4/3	5/3	6/3	(n+2)/3
NOR		5/3	7/3	9/3	(2n+1)/3

El retardo debido a las cargas parásitas también puede ser estimado por medio del modelo RC, en el caso del inversor posee 3 unidades de capacitancia en su salida, por lo que su retraso parásito contribuye $3RC = \tau$, por lo que normalizando este retardo es igual a 1. En la tabla 2.2 están los retardos parásitos de las compuertas AND y OR para varias entradas.

Tabla 2.2: Retardos parásitos de compuertas lógicas de compuertas lógicas.

Compuerta	Número de entradas				
	1	2	3	4	n
Inversor	1				
NAND		2	3	4	n
NOR		2	3	4	n

2.4.4. Retardo en redes lógicas multietapa

Para aplicar los modelos de retardo y lograr dimensionar los anchos de los transistores de forma eficiente es necesario conocer el camino lógico crítico de la función lógica a implementar, de modo de que si este camino cumple con los requerimientos de tiempo, todo el diseño cumplirá con las especificaciones.

El esfuerzo lógico de camino G, se expresa como el producto de los esfuerzos lógicos de cada etapa a través del camino.

$$G = \prod g_i \tag{11}$$

El esfuerzo eléctrico de camino H, se expresa como la capacitancia de salida del camino dividida entre la capacitancia de entrada del camino. Es más conveniente definir este esfuerzo como una relación de capacitancias del camino que como el producto de los diferentes esfuerzos de cada etapa individual, ya que se desconocen hasta que los anchos sean dimensionados.

$$H = \frac{C_{out(camino)}}{C_{in(camino)}} \tag{12}$$

El esfuerzo de camino F, es el producto de los esfuerzos de cada etapa

$$F = \prod f_i = \prod g_i h_i \tag{13}$$

También es necesario agregar un nuevo esfuerzo para los caminos que tiene ramificaciones, llamado esfuerzo de ramificación b, el cual se expresa como la capacitancia total vista desde la etapa sobre la capacitancia de la siguiente etapa en el camino

$$b = \frac{C_{total}}{C_{camino}} \tag{14}$$

De las misma forma que existe el esfuerzo de bifurcación o ramificación de etapa, existe una de camino, que se define como el producto de los esfuerzos de camino de las etapas.

$$B = \prod b_i \tag{15}$$

Debido a que el esfuerzo de camino F cambia si hay ramificaciones, se define un esfuerzo general que incluye esta posibilidad, que se presenta en la siguiente ecuación:

$$F = GBH \tag{16}$$

Ahora es posible calcular el retraso en redes de varias etapas, donde el retraso de camino D es la suma de los retardos de cada etapa, igualmente se puede escribir como la suma de los retardo de esfuerzo de etapa D_F y los retardos parásitos del camino P.

$$D = \sum d_i = D_F + P \qquad D_F = \sum f_i \quad P = \sum p_i \tag{17}$$

Como el esfuerzo de retardo de camino es la suma de los esfuerzos de las etapas individuales, y se sabe que la suma de una serie de números cuya suma es constante, puede ser minimizada escogiendo todos los números iguales, esto significa que el retardo del camino puede ser minimizado cuando cada etapa posee el mismo esfuerzo. Entonces si el camino tiene N etapas y deben tener el mismo esfuerzo, se debe cumplir lo siguiente:

$$\hat{f} = g_i h_i = F^{1/N} \tag{18}$$

Así el retardo mínimo posible de un camino de N etapas con esfuerzo de camino F y retardo parásito de camino P es:

$$D = NF^{1/N} + P \tag{19}$$

El resultado anterior es la clave para tener un dimensionamiento adecuado respecto al retardo, por lo que si se utiliza la definición de la ecuación (7) combinada con la ecuación (10) se puede derivar una fórmula para encontrar la capacitancia de entrada adecuada a su carga [7] y a partir de esta su ancho, como se muestra en la ecuación (20).

$$C_{in_i} = \frac{C_{out} \times g_i}{\hat{f}} \tag{20}$$

El proceso de despeje empieza desde el final del camino, hasta llegar a la primera compuerta aplicando la ecuación (20), así determinando el ancho de la compuerta. Luego manteniendo la relación entre transistores NMOS y PMOS, se asigna los respectivos tamaños que debe aportar cada uno, cumpliendo el total en la entrada de la etapa.

3 Diseño y desarrollo

En esta sección se detalla el proceso de diseño y pruebas realizadas para la obtención del módulo de decodificación fila para la memoria. El diseño incluye tanto consideraciones de área en función de la tecnología CMOS a usar, así como el dimensionamiento y verificación de cada uno de los submódulos que conformar el decodificador, así como el bloque ya conectado al arreglo de celdas SRAM.

3.1. Decodificador de fila

Uno de los bloques funcionales más importantes en circuitos digitales son los decodificadores, se encuentran en cualquier circuito de selección, usados para construir funciones lógicas. En el caso de las memorias, son necesarios decodificadores para controlar el acceso a las celdas que almacenan la información, acomodadas en arreglos de fila y columna. La figura 3.1 describe las entradas y salidas típicas de un decodificador N por M.

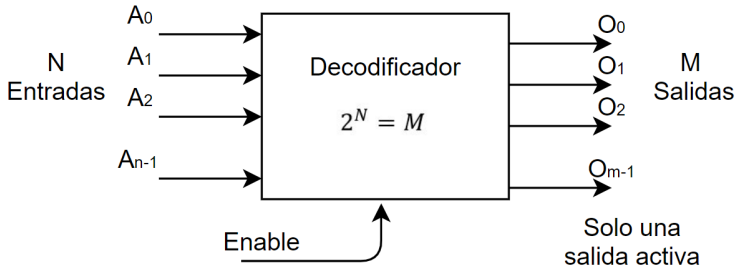


Figura 3.1: Diagrama general de bloque de decodificador a diseñar.

Debido a la gran cantidad de celdas de almacenamiento en las memorias actuales, existen diferentes soluciones orientadas a la reducción del consumo de potencia y mejoras en el desempeño de decodificadores. En función de las especificaciones que se detallarán más adelante, en el caso de este proyecto se optó por un decodificador jerárquico con predecodificación.

El modo más directo para construir un decodificador con *enable* consiste en la implementación del producto de mintérminos (AND booleana), tal como se muestra en la figura 3.2.

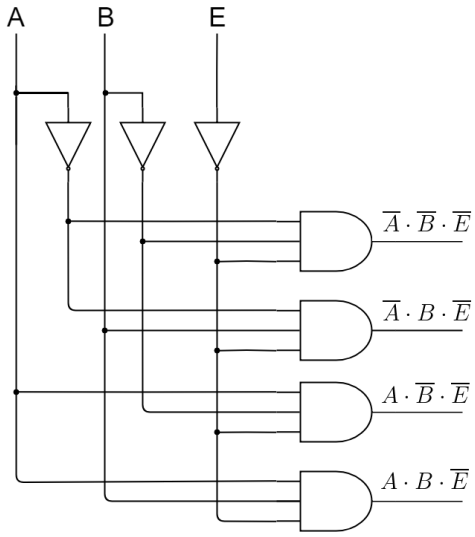


Figura 3.2: Diagrama de decodificador con compuertas ANDS.

Varias de las grandes desventajas que tiene este diseño son su dependencia al crecimiento en área de las compuertas AND respecto a la cantidad de entradas. Si el decodificador que se requiere es de 6x64 bits más la entrada del *enable*, ello significa un total de 64 compuertas AND de 7 entradas cada una, lo que implica catorce transistores en la NAND de entrada, más dos del inversor de salida. Ello implica no solo área extra en una sola compuerta, sino que conlleva también el efecto de muchos transistores en serie que penalizan el esfuerzo lógico de una compuerta NAND (según explica [7]).

Si se utiliza la pre-decodificación para dividir la carga en más etapas y se limita la cantidad de entradas de las compuertas a solamente dos, se puede evaluar como de punto de partida el circuito del decodificador de 3x8 bits de la figura 3.3.

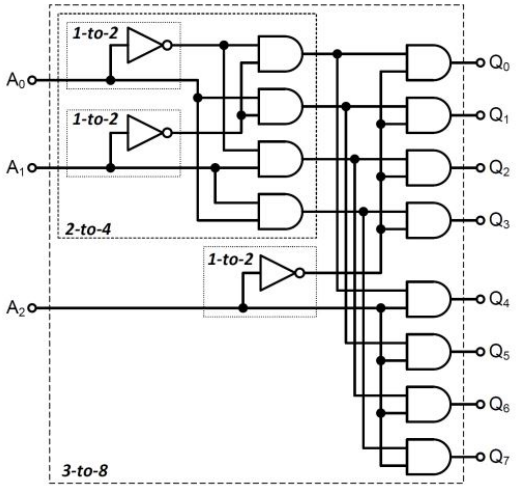


Figura 3.3: Decodificador de 3x8 bits compuertas ANDs con pre-decodificación.

Pese a la mejor distribución de las compuertas en varias etapas, aún está el problema de usar compuertas ANDs, debido a la lógica CMOS, solamente se tienen compuertas NAND, NOR y NOT, y es necesario compuertas para generarla. Se pretende construir solo con estas compuertas el decodificador para reducir la cantidad de transistores.

A partir de la función de decodificación de n entradas se tiene la siguiente ecuación:

$$m_i = l_0 \cdot l_1 \cdot l_2 ... l_{n-1} \tag{21}$$

Donde l_0, l_1, ... ,l_{n-1} son las variables de entrada o su complemento y m_i es el mintérmino al que corresponden las entradas. Según el teorema de De Morgan es posible transformar la ecuación (21) de tal forma que la implementación de los mintérminos sea descrita con compuertas NOR y NAND [11]. Así los productos pueden cambiarse por una suma lógica en complemento, como se expresa a continuación:

$$a \cdot b = \overline{\overline{a} + \overline{b}} \tag{22}$$

Empezando por la última variable l_{n-1}, usando la doble negación y la ecuación (22) se puede obtener progresivamente una ecuación con compuertas NAND y NOR alternadas y conectadas en serie, tal como se muestra en la ecuación (24).

$$m_i = l_0 \cdot l_1 \cdot l_2 ... l_{n-1} = \overline{\overline{l_0 \cdot l_1 \cdot l_2 ... l_{n-4} \cdot l_{n-3} \cdot l_{n-2} \cdot l_{n-1}}} \tag{23}$$

$$= \overline{\overline{\overline{l_0 \cdot l_1 \cdot l_2 ... l_{n-4} \cdot l_{n-3} \cdot l_{n-2} \cdot l_{n-1}} + \overline{l_{n-1}}} + \overline{l_{n-1}}} = \overline{\overline{\overline{l_0 \cdot l_1 \cdot l_2 ... l_{n-4}} + \overline{l_{n-3}} \cdot l_{n-2}} + \overline{l_{n-1}}} \tag{24}$$

Luego de simplificar cualquier ecuación se obtiene un decodificador conformado únicamente de compuertas de dos entradas y siempre de n-1 etapas. Es posible con la ecuación (24) detener la simplificación antes para lograr una menor cantidad de etapas pero esto supone una primera etapa de compuertas con mayor cantidad de entradas.

Se presenta a modo de ejemplo la simplificación para la ecuación de seis variables del decodificador a realizar.

$$\overline{\overline{\overline{\overline{l_0} + \overline{l_1} \cdot l_2} + \overline{l_3} \cdot l_4} + \overline{l_5}} \tag{25}$$

Como se observa de la ecuación anterior, la primera etapa siempre está compuesta de un decodificador de 2x4 bits y dependiendo si la cantidad de entradas o variables es par, la primera etapa estará basada en NORs y si son impares la primera etapa estará basada en NANDs, como se nota para las variables l_0 y l_1. Nótese también que en el caso de las operaciones NOR las variables están negadas y en el caso de las NAND; esto lleva a la necesidad de colocar un inversor en bloques de NANDs y NORs. Para las etapas de NANDs el inversor debe ser ubicado para la mitad de bits de salida menos significativa y para las etapas de NORs la mitad de bits de salida más significativa.

3.1.1. Diagramas lógicos de los bloques base del decodificador de fila

Bajo los criterios expuestos en la sección anterior, se procedió a diseñar cada uno de los sub-bloques con los que se construiría el decodificador final. Los bloques de la primera etapa son llamados BASE_NOR y BASE_NAND, y son decodificadores 2x4 bits como se muestran en la figura 3.4

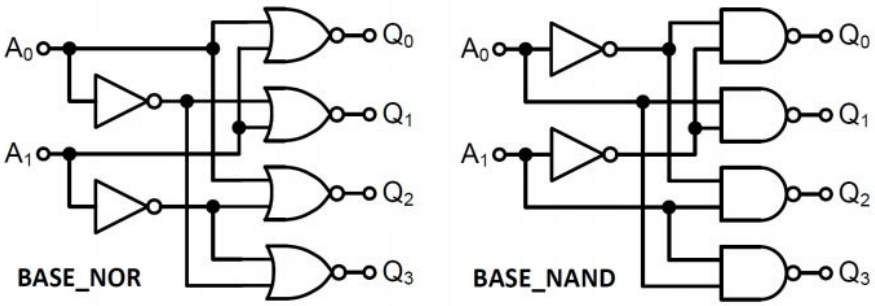


Figura 3.4: Diagrama de BASE_NOR y BASE_NAND a nivel de compuerta.[11]

Los siguientes bloques son llamados NANDS y NORS, los cuales están compuestos de cuatro compuertas NANDs y cuatro NORs respectivamente y se muestran en la figura 3.5.

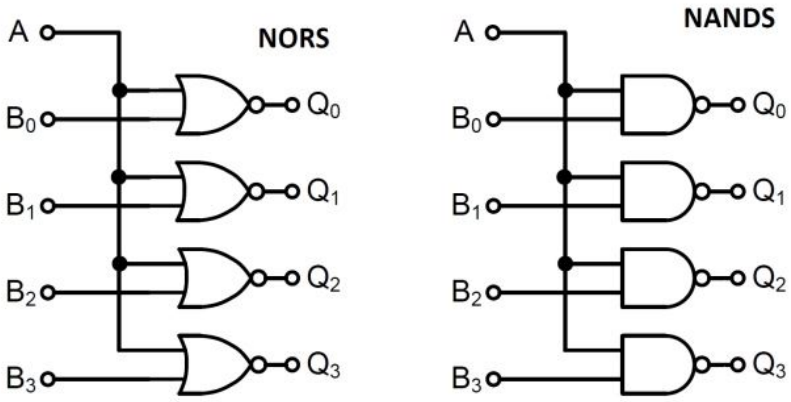


Figura 3.5: Diagrama de NANDS y NORS a nivel de compuerta.[12]

Y los últimos bloques son el NNANDS y el NNORS que se componen de grupos de cuatro de sus respectivas compuertas y un inversor, mostrados en la figura 3.6

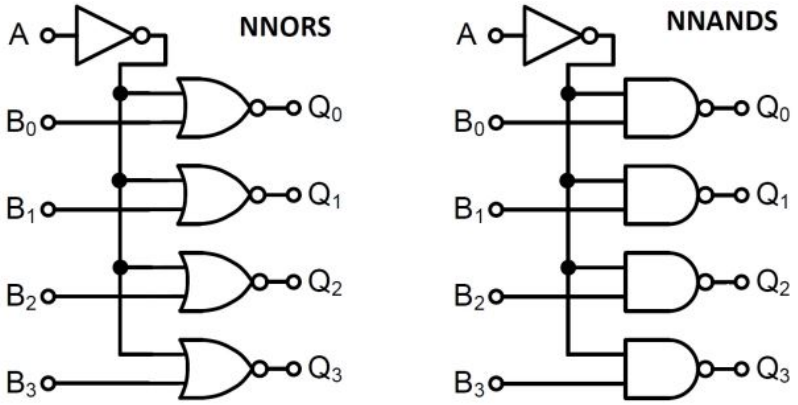


Figura 3.6: Diagrama de NNANDS y NNORS a nivel de compuerta.[12]

El diseño de cualquier tamaño de decodificador a nivel esquemático es simple usando los bloques descritos anteriormente con los principios de conexión y ubicación. Un detalle importante es que a nivel funcional solo se necesita de un inversor para los bloques de las terceras etapas en adelante para los respectivos bits que requieren negación, pero igualmente los bloques individuales se construyen conteniendo este inversor para la simplificación del cableado.

A continuación se presenta el diagrama de bloques del decodificador de 6x64 bits conformado por los sub-bloques anteriores.

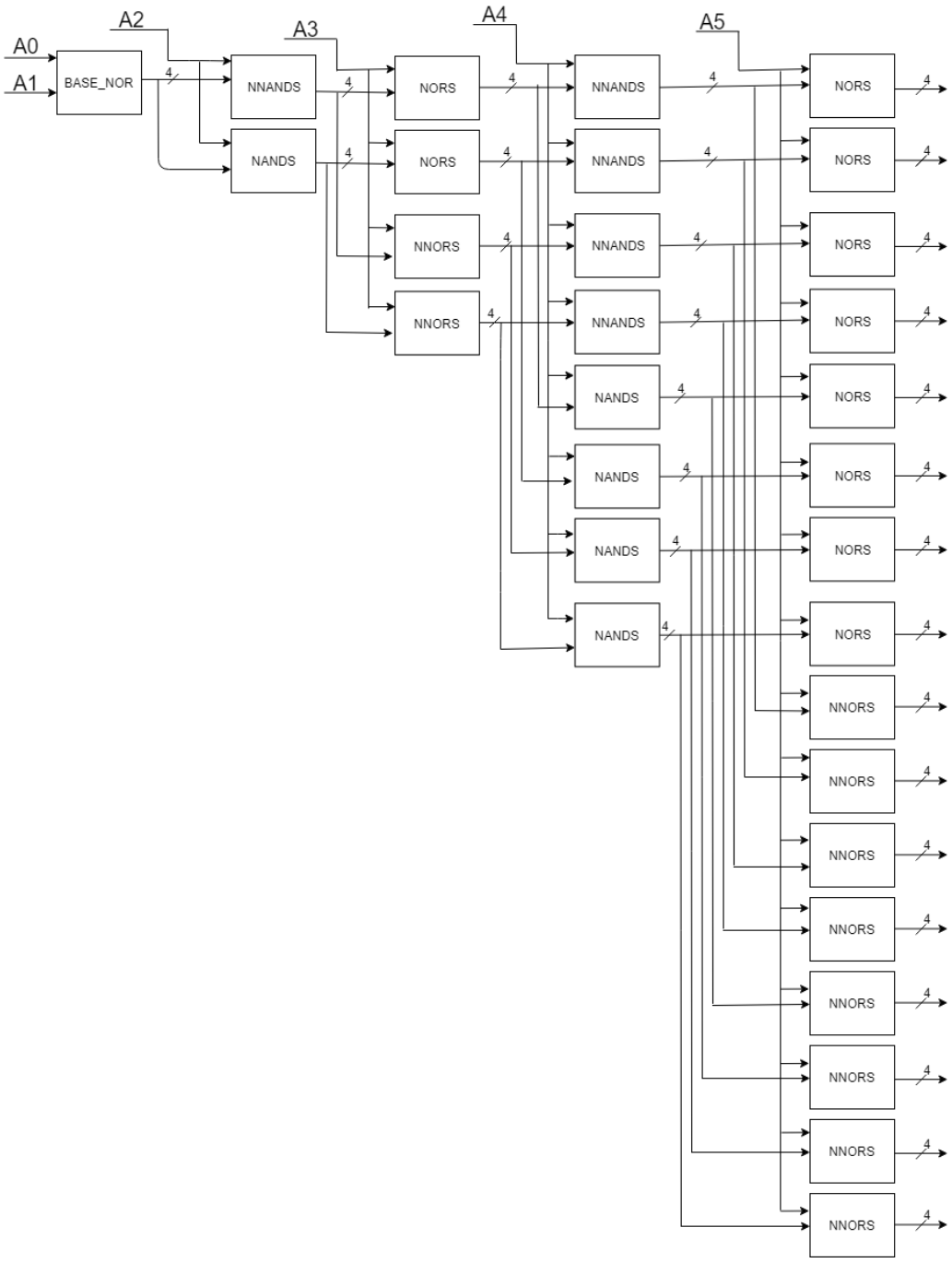


Figura 3.7: Diagrama de bloques del decodificador de 6x64 bits a partir de sub-bloques.

Con ayuda de la herramienta Custom Compiler se conecta a nivel de compuerta la estructura del decodificador de la figura 3.7 para hacer una simulación únicamente lógica. El comportamiento obtenido concuerda con el esperado de la tabla de verdad de un decodificador (por cuestiones de tamaño de la tabla de resultados no se muestra la tabla ya que son 64 posibles combinaciones).

3.2. Circuito de la etapa de habilitación

Para el circuito de la etapa de habilitación o *enable* del decodificador resulta conveniente diseñar de forma conjunta esta etapa y la de *buffers*, para que ambas etapas sean contempladas en el estudio de los caminos críticos para un posterior dimensionamiento de todas las compuertas que conforman todos los sub-bloques.

A continuación se presenta el comportamiento deseado con la tabla de verdad de la etapa, a partir de la cual se obtendrá el circuito combinacional requerido que cumple el rol de esta etapa.

Tabla 3.1: Tabla de verdad de etapa de habilitación.

Enable	**Dato**	**Salida**
0	0	0
0	1	0
1	0	0
1	1	1

De la tabla 3.1 se tiene que las señales de *Enable* y *Dato* son las entradas de este bloque y la señal de *Salida* es el resultado requerido, se observa de la tabla anterior que el comportamiento es el de una compuerta AND. Es posible ahorrar un inversor de la siguiente etapa de *buffers* o manejadores de línea, de tal modo que se utilice una etapa de NANDs conectada en serie con un solo inversor en vez de una cadena, siendo este inversor el *buffer* que debe maneje la capacitancia del cable de *wordline* de la memoria.

En el siguiente diagrama se muestra el decodificador de 6x64 bits con la señal y la respectiva etapa de habilitacion incluida, donde éste incorpora el inversor de salida como un solo sub-bloque funcional en una compuerta AND por cada una de las 64 señales de salida del decodificador.

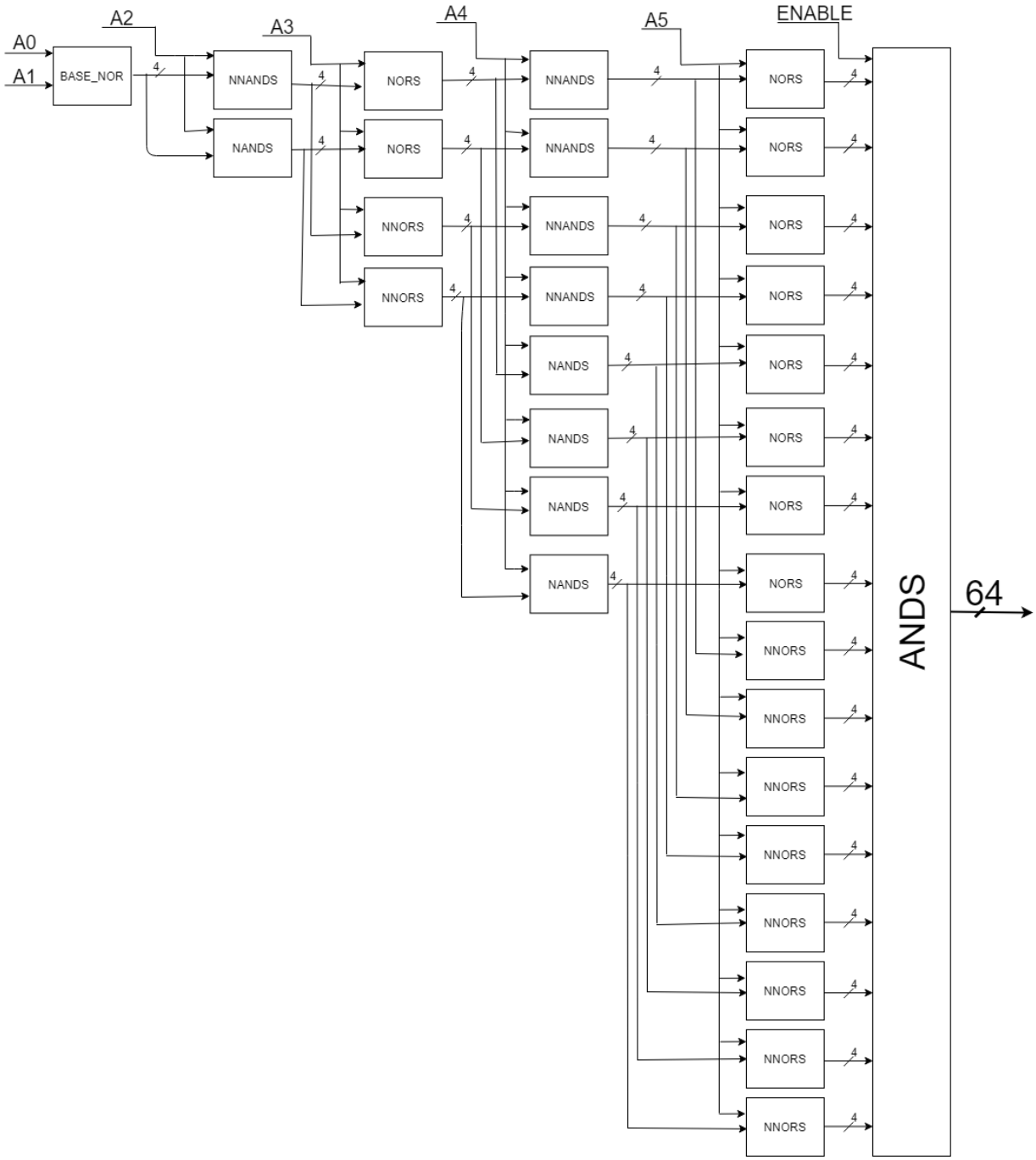


Figura 3.8: Diagrama de bloques del decodificador de 6x64 bits con *enable* a partir de sub-bloques.

El diagrama de bloques de la figura 3.8 al tener ya integradas las funcionalidades del decodificador y el *enable*, es necesario realizar un estudio en el tiempo de retardo de las señales desde las entradas hasta la salidas con la respectiva carga que ejerce la memoria para determinar cual de los diferentes caminos es el que posee mayor retardo y recorre todas las etapas de compuertas. Para esto es necesario conocer de antemano la capacitancia de *wordline*. Por esta razón se realiza una simulación con las respectivas extracciones de resistencia y capacitancia parásita de las interconexiones entre metales y transistores sobre una celda de memoria de un bit. La vista de la celda de trazado se muestra en la figura 3.9.

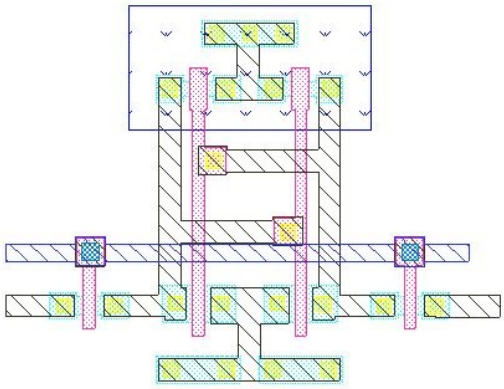


Figura 3.9: Trazado del bloque fundamental de bit del arreglo de celdas de memoria SRAM a manejar por el decodificador. .

Con el fin de obtener la capacitancia, existe una opción que se encuentra en las herramientas de simulación, que permite observar todas las capacitancias conectadas a un nodo de interés (en este caso el METAL2 de color amarillo en la figura 3.9 de la *wordline*).

Sumando todas estas capacitancias que suponen una carga en este nodo, se tiene una capacitancia aproximada C_{gcelda} (cuyo valor no se puede revelar por estar los datos de la tecnología protegidos por un acuerdo de confidencialidad).

Con este valor se calcula la capacitancia de carga que debe manejar el decodificador por *wordline*. Así si hay 32 celdas por *wordline* [2], la capacitancia es aproximadamente 32 veces C_{gcelda}.

Para simular los caminos críticos se usan las compuertas mínimas dependiendo de ya sea NAND, NOR o NOT de la figura 3.10. Además se unifican las compuertas idénticas en una sola compuerta que emule la carga que ejercen todas las interconectadas al nodo y finalmente se coloca el capacitor en la salida para simular la carga que ejerce la *wordline*, con el objetivo de que el circuito sea sencillo de interconectar en el ambiente del simulador.

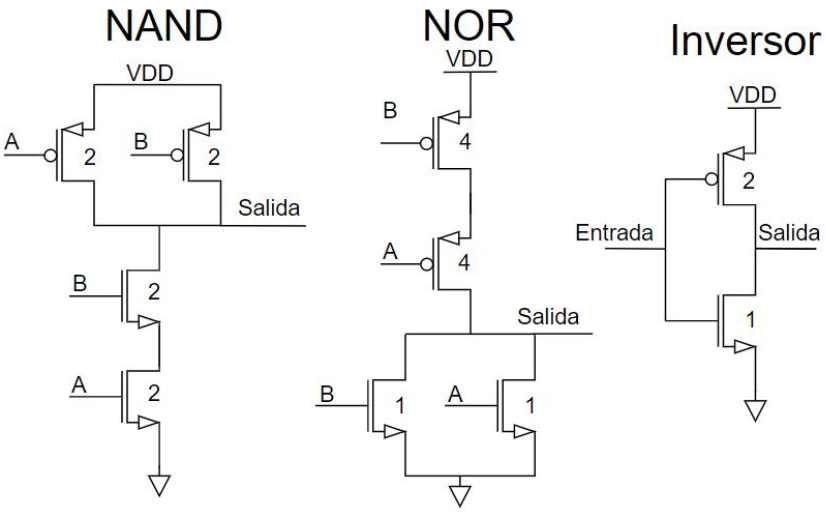


Figura 3.10: Diagrama a nivel de transistor de compuertas NAND, NOR e Inversor.

Los tamaños de los transistores PMOS y NMOS de la figura 3.10 son para obtener un valor igual a R en las transiciones, por esta razón los transistores PMOS en la compuerta NAND tiene un ancho de 2 ($2R/2 = R$) y los transistores NMOS en serie tiene ancho 2 para un total de $R/2 + R/2 = R$. De forma similar en la compuerta NOR los transistores PMOS en serie tienen un ancho de 4 para un valor de $2R/4 + 2R/4 = R$ y por último el inversor tiene resistencia $2R/2$ y $R/1$ para los transistores PMOS y NMOS respectivamente. En los diagramas de los caminos críticos se multiplican el tamaño de las compuertas antes descritas por el factor indicado para reunir varias compuertas en un sola y simplificar el circuito, mientras se mantienen las cargas.

Es importante mencionar que las siguientes simulaciones son un punto de partida en el estudio de los retardos, basado en la construcción a nivel de compuerta del decodificador. En donde los tamaños son establecidos por un mínimo o una escala basado en ese mínimo de los tamaños de los transistores de las compuertas NAND, NOR y NOT antes mostradas.

El camino uno se establece desde la entrada del nodo de *enable* hasta la salida. Cabe notar que es indiferente cuál de las 64 posibles salidas se tome, pues la carga es la misma y por ende el camino posee el mismo *fanout*. En la figura 3.11 se muestra el circuito equivalente del camino uno.

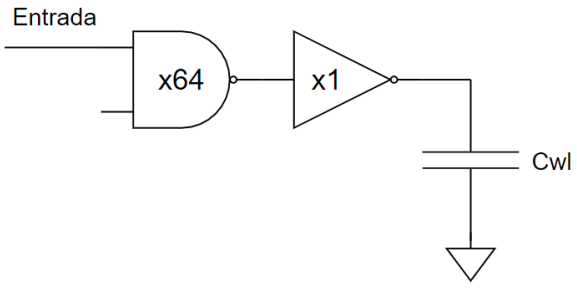


Figura 3.11: Diagrama de compuertas equivalentes del camino uno.

El nodo de la señal *enable* está conectado con 64 compuertas NAND, luego las salida de cualquiera de estas compuertas se encuentra conectada en serie con un inversor. El correspondiente ancho de los transistores se presenta en la tabla 3.2.

Tabla 3.2: Anchos de transistores del camino uno.

	NAND	**Inversor**
PMOS (μm)	28,16	0,44
NMOS (μm)	28,16	0,22

El camino dos es desde el bit de dirección menos significativo ($A0$) del decodificador hasta la salida. Este camino se muestra en la figura 3.12:

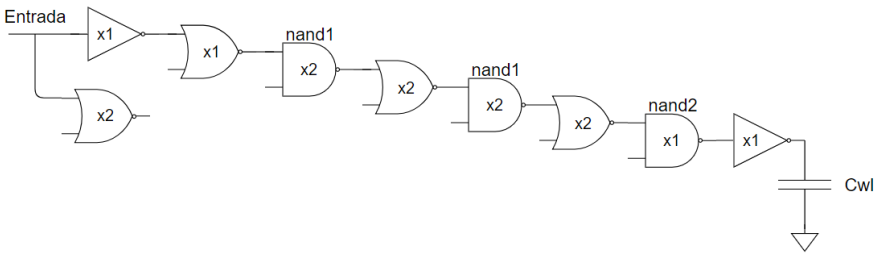


Figura 3.12: Diagrama de compuertas equivalentes del camino dos.

En la tabla 3.3 presenta los tamaños de los transistores usados en la simulación.

Tabla 3.3: Anchos de transistores del camino dos.

	Inversores	**NORS**	**NAND1**	**NAND2**
PMOS (μm)	0,44	1,76	0,88	0,44
NMOS (μm)	0,22	0,44	0,88	0,44

El camino tres es desde el bit $A4$ hasta la salida y posee las compuertas interconectadas como se muestra en la figura 3.13.

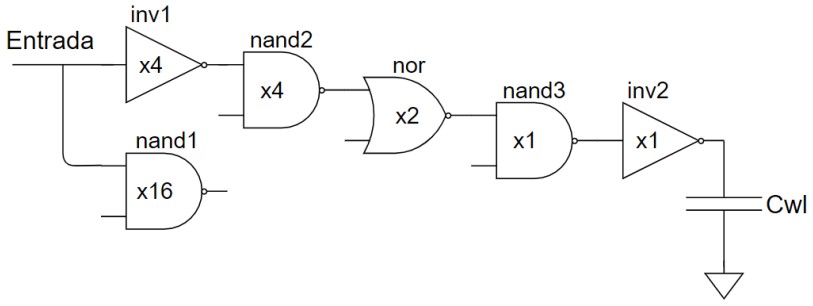


Figura 3.13: Diagrama de compuertas equivalentes del camino tres.

En la tabla 3.4 se presentan los tamaños de los transistores usados en la simulación.

Tabla 3.4: Anchos de transistores del camino tres.

	Inv1	**Inv2**	**NOR**	**NAND1**	**NAND2**	**NAND3**
PMOS (μm)	1,76	0,44	1,76	7,04	1,76	0,44
NMOS (μm)	0,88	0,22	0,44	7,04	1,76	0,44

El camino cuatro es desde el bit $A5$ hasta la salida y posee las compuertas interconectadas como se muestra en la figura 3.14

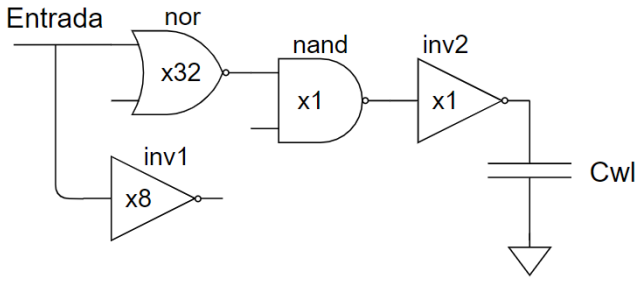


Figura 3.14: Diagrama de compuertas equivalentes del camino cuatro.

En la tabla 3.5 se presentan los tamaños de los transistores usados en la simulación.

Tabla 3.5: Anchos de transistores del camino cuatro.

	Inv1	**Inv2**	**NAND1**	**NOR**
PMOS (μm)	3,52	0,44	0,44	28,16
NMOS (μm)	1,76	0,22	0,44	7,04

El camino cinco es desde el bit $A2$ hasta la salida y su circuito es el mostrado en la figura 3.15

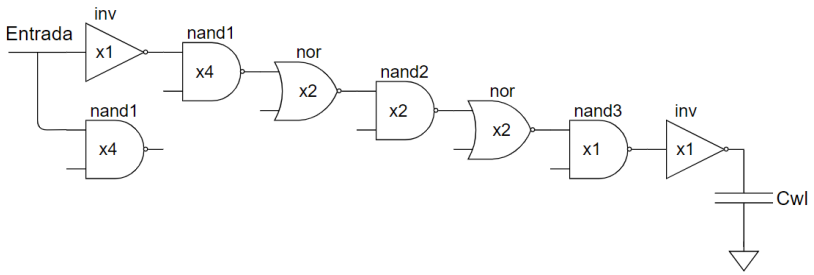


Figura 3.15: Diagrama de compuertas equivalentes del camino cinco.

En la tabla 3.6 se presenta los tamaños de los transistores usados en la simulación.

Tabla 3.6: Anchos de transistores del camino cinco.

	Inversor	**NAND1**	**NAND2**	**NAND3**	**NOR**
PMOS (μm)	0,44	1,76	0,88	0,44	1,76
NMOS (μm)	0,22	1,76	0,88	0,44	0,44

El camino seis es desde el bit $A3$ hasta la salida y posee el camino lógico de compuertas interconectadas como se muestra en la figura 3.16:

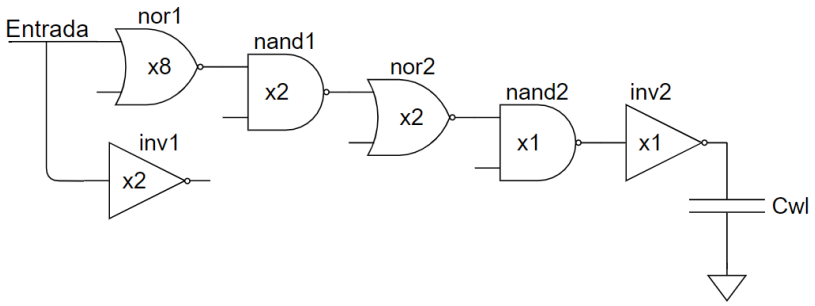


Figura 3.16: Diagrama de compuertas equivalentes del camino seis.

En la tabla 3.7 se encuentran los tamaños de los transistores usados en su simulación.

Tabla 3.7: Anchos de transistores del camino seis.

	Inv1	**Inv2**	**NOR1**	**NAND1**	**NOR2**	**NAND2**
PMOS (μm)	0,88	0,44	7,04	0,88	1,76	0,44
NMOS (μm)	0,44	0,22	1,76	0,88	0,88	0,44

Todos los caminos anteriores se simularon de tal forma que se pueda observar las transiciones en la carga y se realizaron varias mediciones en su tiempo de propagación para encontrar un valor promedio, los cuales se presentan en la tabla resumen 3.8.

Tabla 3.8: Tiempos de propagación promedio de los caminos críticos.

Camino	t_{pd}**(ns)**
1	0,706
2	1,16
3	0,903
4	0,785
5	1,17
6	0,929

Según la tabla anterior, los caminos dos y cinco son los caminos que poseen mayor retardo con una diferencia de camino de 10 ps. Es importante indicar que ambas simulaciones son aproximaciones dado los anchos y la agrupación de compuertas conectadas al mismo nodo, con el fin de imitar la capacitancia total que debe manejar la compuerta anterior. Pero el camino que cruza todas las compuertas de inicio a fin el diseño, que permite dimensionar todo el diseño, es el camino 2, por esta razón se escoge este para optimizar.

3.4. Dimensionamiento óptimo del decodificador con circuito de enable

Una vez escogido el camino crítico a dimensionar es necesario conocer algunas características de las tecnología de fabricación con el objetivo de normalizar tamaños en base a la capacitancia de compuerta para simplificar cálculos posteriores.

Con la capacitancia de transistor mínimo de la tecnología, se normaliza la capacitancia de la *wordline* para realizar los cálculos de esfuerzo lógico y dimensionamiento de las compuertas del camino critico seleccionado. Para calcular esta capacitancia en una versión más pesimista que incluye el peor de los casos donde los transistores conmutan entre corte y triodo. se utiliza aquí la ecuación recomendada por el fabricante al que se enviaría el diseño, la cual es la que proporciona las herramientas y los datos del proceso de fabricación.

$$C_{gs} = W_{dib}L_{dib}C_{OX} + W_{dib}C_{OV} \tag{26}$$

Donde C_{gs} es la capacitancia entre las terminales de la compuerta y el surtidor, W_{dib} es el ancho del transistor, L_{dib} es el largo del transistor, C_{OX} es la capacitancia del óxido por micrómetro y C_{OV} es la capacitancia de traslape por micrómetro. Por motivos de confidencialidad no se muestran los datos propios del proceso de fabricación ni las capacitancias resultantes del cálculo.

Luego para tener una sola capacitancia equivalente en la entrada del camino, se juntan las capacitancias que aportan las compuertas NOR y el inversor, ya que es necesario tener de antemano esta capacitancia para realizar los cálculos de esfuerzo lógico. Además arbitrariamente se escogen de tamaño mínimo para generar la menor carga posible a la etapa de *latches* anterior al decodificador, que posteriormente en esta sección se diseñará. El diagrama final se ofrece en la figura 3.17.

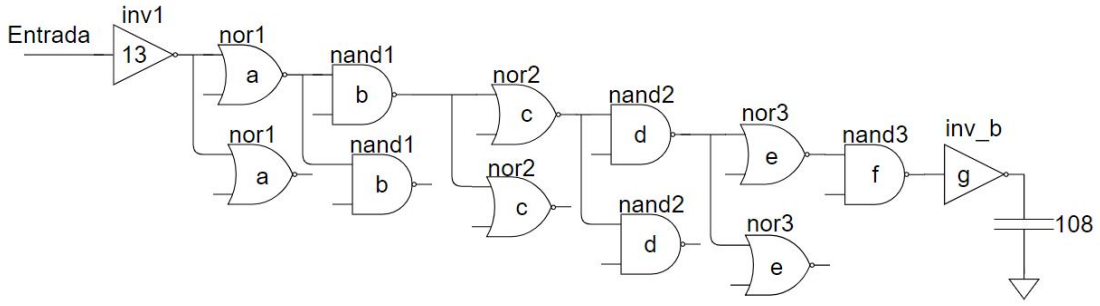


Figura 3.17: Diagrama del camino crítico a dimensionar.

Para el circuito anterior se calcula el esfuerzo lógico, eléctrico y de ramificación de camino para luego llegar al esfuerzo de camino más eficiente.

$$G = 1 \times \frac{5}{3} \times \frac{4}{3} \times \frac{5}{3} \times \frac{4}{3} \times \frac{5}{3} \times \frac{4}{3} \times 1 = \frac{8000}{729}$$
$$B = 2^5 = 32$$
$$H = \frac{108}{13}$$
$$F = G \times B \times H \approx 2917{,}38$$
$$\hat{f} = \sqrt[8]{2917,38} \approx 2,11$$

A partir del esfuerzo mínimo obtenido y usando la ecuación (20), se despejan los anchos de todas las etapas que conforman el circuito del decodificador y el del *enable*. Los valores se presentan en la tabla 3.9 según la denominación de la figura 3.17.

Tabla 3.9: Anchos de transistores normalizados del camino crítico.

Compuerta	C_{in}	**PMOS**	**NMOS**
inv_b	51,18	34,12	17,06
nand3	32,34	16,17	16,17
nor3	25,54	20,432	5,108
nand2	16,14	8,07	8,07
nor2	12,75	10,2	2,55
nand1	8	4	4
nor1	6,32	5,056	1,26
inv1	13	8,66	4,33

Los datos de la tabla anterior se desnormalizan a su valor físico real y simultáneamente, se introducen en la herramienta para visualizar que tamaños son los más cercanos a los anteriores, ya que existe cierta resolución en los anchos que pueden ser fabricados, en la tabla 3.10 se muestran esos valores ya redondeados por la herramienta. Para efectos de este proyecto se utilizaran los tamaños determinados en la tabla anterior por cuestiones de tiempo, pero es posible a partir del uso de un algoritmo genético que parta de estos tamaños iniciales encontrados, tal como el descrito en [16, 17, 18] para encontrar tamaños óptimos para cada compuerta en función de su situación en la ruta crítica. Incluso con este método podría también explorarse la topología misma del decodificador. Esta opción debe valorarse en futuras iteraciones para disminuir aún más el consumo de potencia del bloque de decodificación.

Tabla 3.10: Anchos físicos de los transistores del camino crítico.

Compuerta	**PMOS** (μm)	**NMOS** (μm)
inv_b	7,505	3,7555
nand3	3,56	3,56
nor3	4,495	1,125
nand2	1,775	1,775
nor2	2,245	0,56
nand1	0,88	0,88
nor1	1,115	0,275
inv1	1,905	0,955

3.5. Diagramas de pretrazado y trazados finales de los circuitos de decodificación de fila y enable

En esta sección se construyen individualmente los pretrazados para tener un primer acercamiento a la distribución y colocación de los transistores, para luego construir los modelos de los trazados en busca de optimizar el área del circuito total. Como ya se indicó, el diseño se compone de seis etapas necesarias para la construcción de la estructura propuesta en la figura 3.8. Siguiendo en el dimensionamiento de la tabla 3.10, se tiene un total de 10 diferentes bloques agrupados. Cabe mencionar que, a pesar de que los pretrazado son una buena representación de los transistores que se desean posteriormente colocar e interconectar, la tecnología impone la necesidad de colocar un contacto en la terminal del substrato de los transistores si se deseara realizar una simulación *postlayout*. Esta terminal se omite en los diagramas pretrazado.

De manera general para todos los pretrazados presentados en este informe, el color gris indica METAL1, el amarillo indica METAL2, el azul indica METAL3, el rojo la difusión, el verde el polisilicio y los cuadros con marcas de equis indican los contactos. Estos colores establecidos son los mismos para el kit de diseño usado en la tecnología CMOS empleada, pero como se invirtieron los colores en las imágenes finales en este documento para mejor visibilidad, se cambian a la siguiente denominación en los trazados físicos ya implementados, el METAL1 es gris, METAL2 azul y el METAL3 es rojo, la difusión es azul claro y el polisilicio es rosado.

3.5.1. BASE_NOR

El circuito a nivel de compuerta de este sub-bloque se mostró en la figura 3.4, conformado por el primer inversor (**inv1**) y las primeras NORS (**nor1**). El comportamiento es el de un decodificador de 2x4 bits y este funciona como la primera etapa de predecodificación. El diagrama pretrazado y trazado final de este sub-bloque se muestran en la figura 3.18.

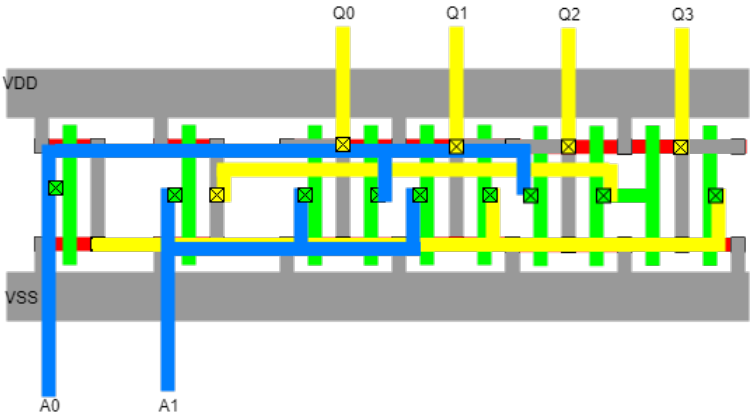


Figura 3.18: Pretrazado de sub-bloque BASE_NOR.

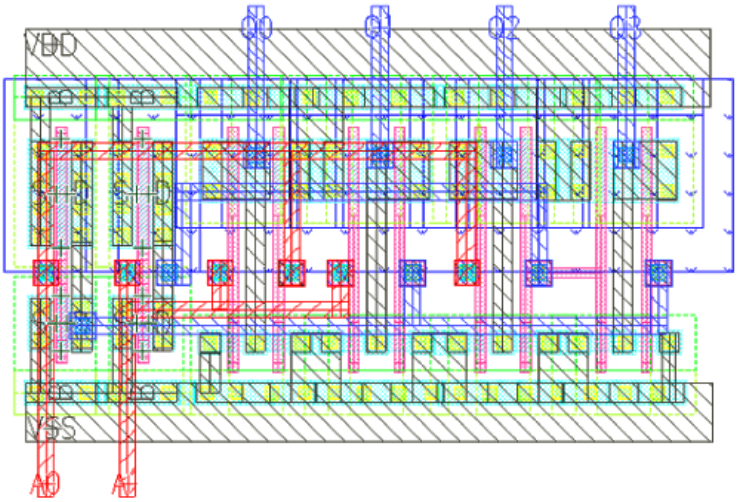


Figura 3.19: Trazado de sub-bloque BASE_NOR.

Los dos inversores se encuentran a la izquierda y las cuatro NORs se encuentran apiladas a la derecha. Se utiliza METAL2 y 3 para para la distribución de las señales de entrada y su complemento para alcanzar todas las compuertas, que se encuentran únicamente en el nivel o capa del METAL1 hacia abajo. Estructuralmente las vista del sub-bloque BASE_NOR es la misma que en el pretrazado pero es notable que el trazado está más junto, además se poseer su respectivos contactos al substrato, los cuales ya se encuentran debidamente conectados a las alimentaciones. También es notorio que el metal sobre el que se distribuyen las alimentaciones son un pocos mas grandes, ya que un mayor tamaño opone menos resistencia y esto supone menos caídas de tensión de alimentación (IR Drop) y en interconexiones no deseadas.

3.5.2. NANDS/NNANDS

El diseño de la figura 3.8 posee dos etapas conformadas por sub-bloques de NANDS y NNANDS. El pretrazado es el mismo por la estructura del circuito a nivel de compuerta (figura 3.5), pero al ser diferentes etapas el ancho es distinto, como se mostró en la tabla 3.10. Las figuras 3.20 y 3.21 muestran los pretrazado de estos sub-bloques.

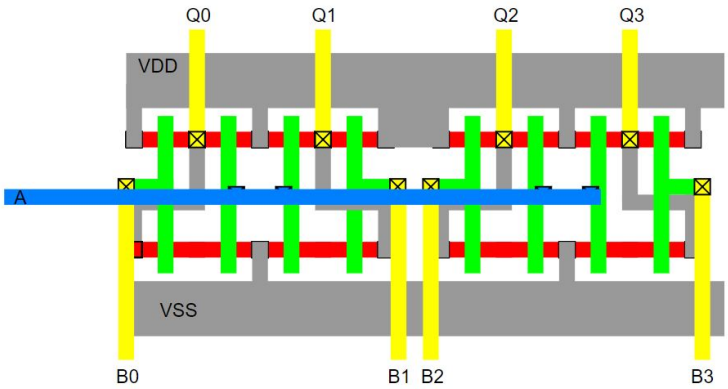


Figura 3.20: Pretrazado de sub-bloque NANDS.

Los diagramas anteriores anterior está conformado por cuatro NANDs apiladas, de modo que se puedan compartir terminales de transistores, tanto para salidas como alimentaciones, por ejemplo en el centro de cualquiera de las dos NANDs laterales de la figura 3.20, donde el contacto de la alimentación es el mismo.

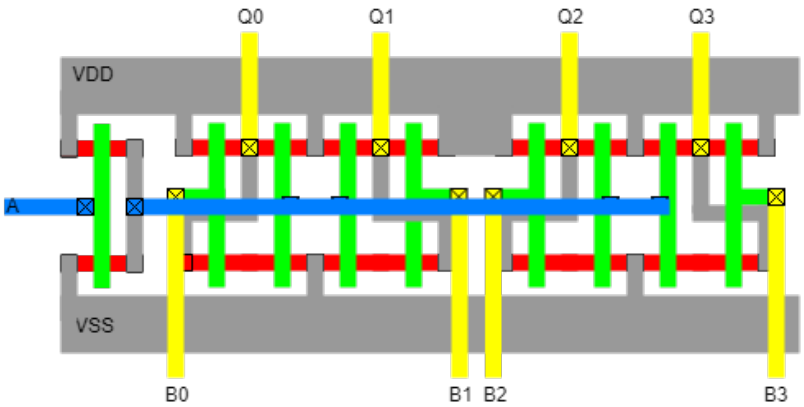


Figura 3.21: Pretrazado de sub-bloque NNANDS.

La estructura del pretrazado de las NNANDS es bastante similar, con la diferencia de que es necesario colocar un inversor antes de la serie de compuertas y la salida de este inversor es una de las entradas comunes en estas compuertas. Los trazados de los sub-bloques **nand1** y **nand2** (NANDS, NNANDS) son los de las figuras 3.22, 3.23, 3.22 y 3.23.

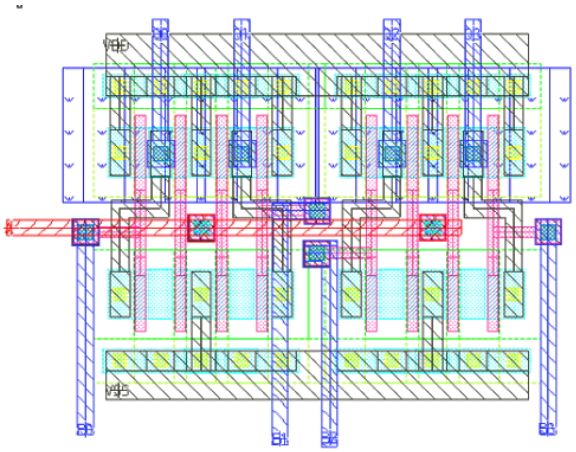


Figura 3.22: Trazado de sub-bloque NANDS de etapa nand1.

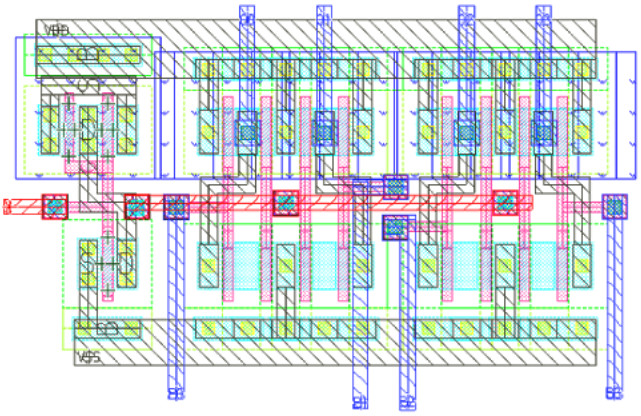


Figura 3.23: Trazado de sub-bloque NNANDS de etapa nand1.

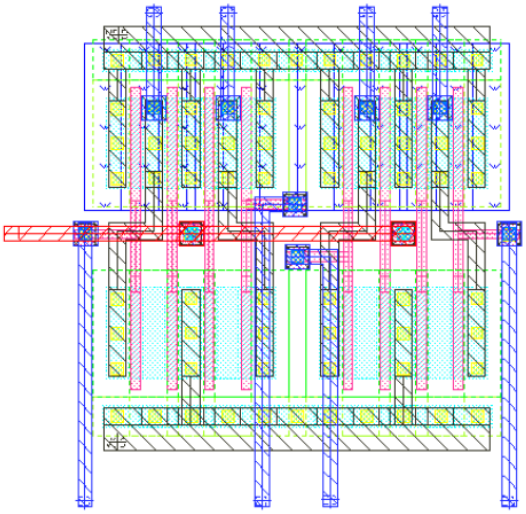


Figura 3.24: Trazado de sub-bloque NANDS de etapa nand2.

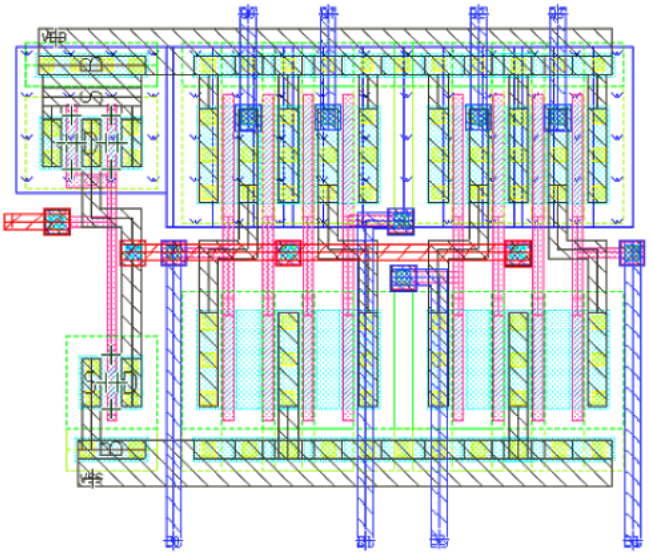


Figura 3.25: Trazado de sub-bloque NNANDS de etapa nand2.

El trazado del bloque NAND se reutiliza para el bloque NAND, con la adición de un simple inversor. Se aprovecha en la medida de lo posible la conexión sin contacto entre difusiones en conexiones series entre transistores, tal como se puede observar en las figuras 3.22, 3.23, 3.22 y 3.23.

Los bloques se compactan entre sí hasta las reglas mínimas de distancia para aprovechar al máximo el espacio o incluso fuerce a sobreponer materiales, como en el caso de los transistores PMOS y la tina de *nwell*, mientras se respete las distancia mínima entre las difusiones, como es el caso de los sub-bloques de NNANDS, entre la primer compuerta NAND y el inversor.

3.5.3. NORS/NNORS

En el caso de los sub-bloques NORS/NNORS, la figura 3.8 también posee dos etapas y los componen las compuertas **nor2** y **nor3** del circuito del camino crítico. Igualmente que las NANDS/NNANDS, la vista pretrazado es la misma, pero cambia para los trazado dependiendo del ancho.

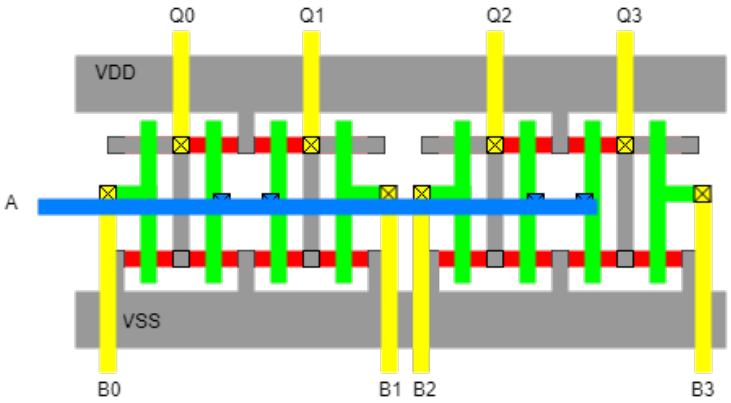


Figura 3.26: Pretrazado de sub-bloque NORS.

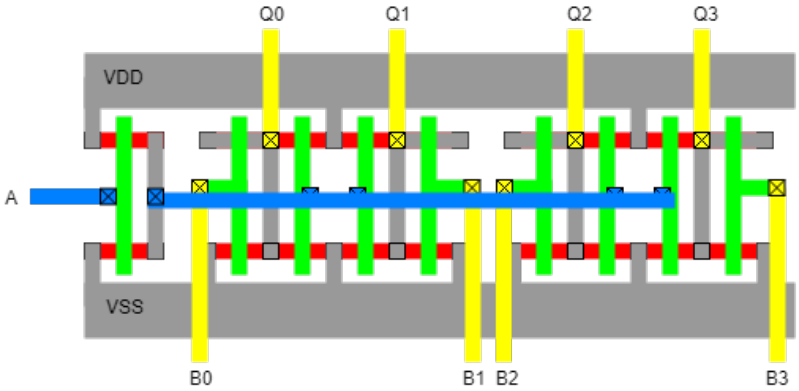


Figura 3.27: Pretrazado de sub-bloque NNORS.

En estos pretrazados, las cuatro compuertas NOR se colocan juntas y se aprovecha la simetría volteando 2 de estas compuertas para compartir algunos puntos en común del circuito a nivel de compuerta. La estructura es muy similar al sub-bloque de las NANDS/NNANDS, pero el orden de los transistores debería intercambiarse entre las redes PMOS y NMMOS. Este cambio se realiza pero no es muy notorio por que las vías que interconectan las capas de metal se sobreponen. El METAL1 al oculta la difusión y la salida en METAL2 disimula que estos transistores están paralelo. El caso del NNORS es el mismo pero se adjunta el inversor de la etapa correspondiente.

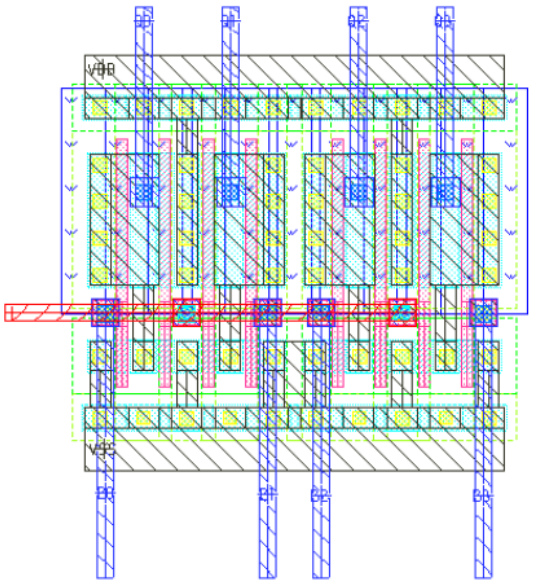


Figura 3.28: Trazado de sub-bloque NORS de etapa nor2.

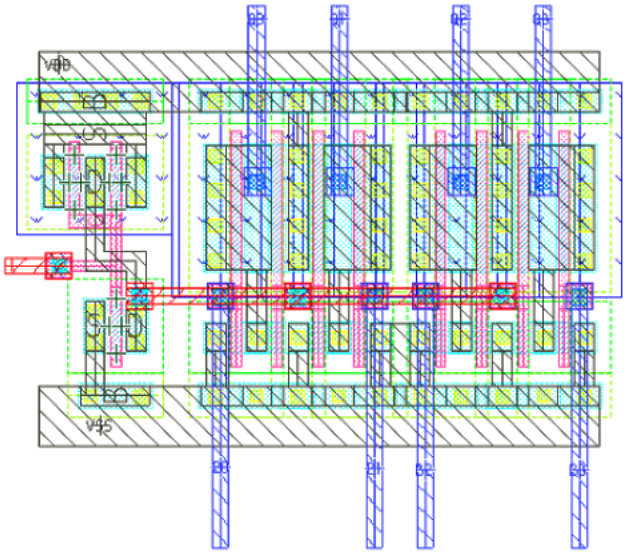


Figura 3.29: Trazado de sub-bloque NNORS de etapa nor2.

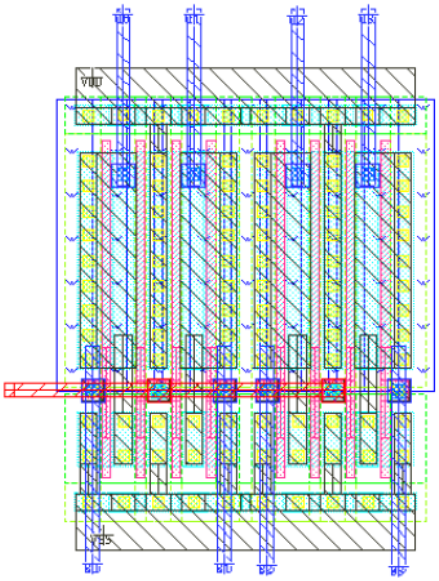


Figura 3.30: Trazado de sub-bloque NORS de etapa nor3.

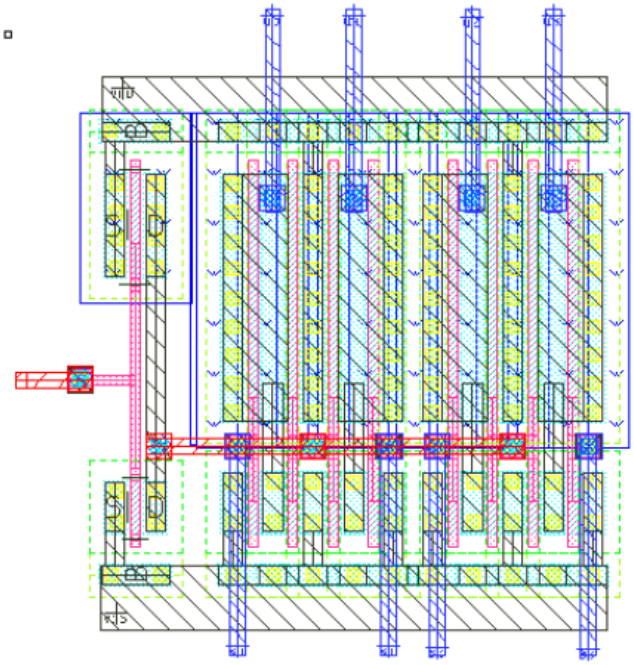


Figura 3.31: Trazado de sub-bloque NNORS de etapa nor3.

3.5.4. ANDS

Ésta es la última etapa del diagrama 3.8 y se compone de dos compuertas, **nand3** que realiza la función del *enable* y este está en cascada con el inversor final **inv_b**, en donde este inversor actúa como *buffer* de salida.

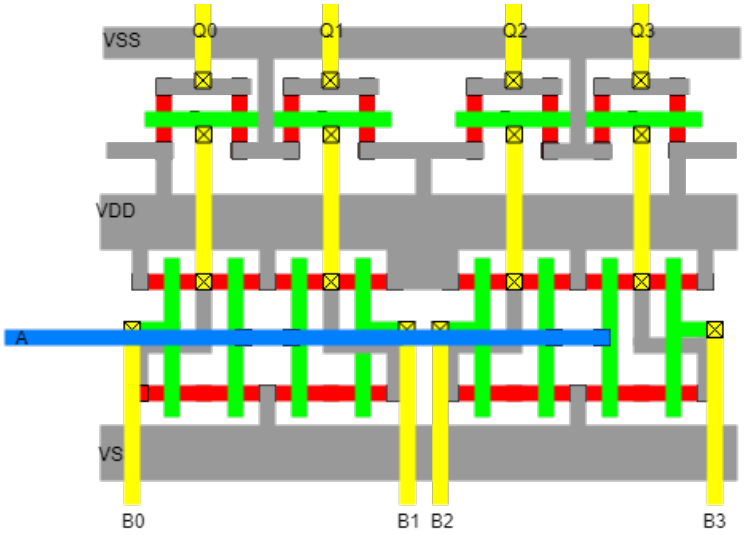


Figura 3.32: Pretrazado de la etapa del *enable-buffers*.

Para este circuito se utiliza la estructura del sub-bloque de NANDS y se colocan cuatro inversores en cada una de las salidas de las NANDs. La conexión de salida pasa de METAL2 por medio de vías hasta el polisilicio de las terminales de compuerta de los transistores. Luego se unifican las alimentaciones y se mantienen las salidas principales en METAL2, ya que las *wordlines* de la memoria también se encuentran en este nivel de metal.

La ventaja de reutilizar la estructura anterior y colocar los inversores en un solo bloque, es el posterior apilamiento de las 64 salidas del decodificador, para un total de 16 ANDS.

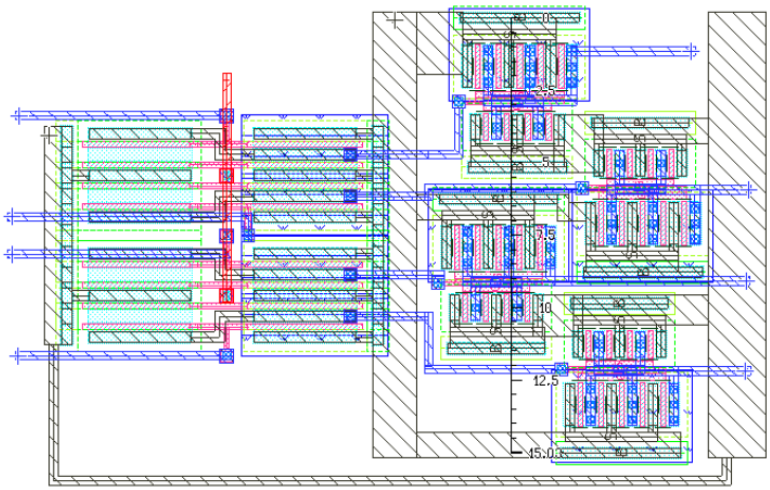


Figura 3.33: Trazado de sub-bloque AND.

La vista de trazado varía un poco sobre la vista preliminar, debido al dimensionamiento necesario de los transistores. Además, el uso de la técnica de dedos (*fingering*) para disminuir los anchos excesivos varía levemente el diseño final.

Esta vista de sub-bloque debe no obstante mejorarse, compactando el área que ocupa.

En la tabla 3.11 se resumen las características fundamentales de los bloques desarrollados hasta el momento.

Tabla 3.11: Características físicas de los sub-bloques desarrollados.

Sub-bloque	**Transistores**	**Alto** (μm)	**Largo**(μm)	**Área** (μm^2)
BASE_NOR	20	5,2	10,92	56,784
NANDS_nand1	16	4,87	6,905	33,63
NNANDS_nand1	18	4,87	9,32	45,3884
NORS_nor2	16	5,145	6,46	33,236
NNORS_nor2	18	5,145	8,645	44,4785
NANDS_nand2	16	6,96	7,035	48,96
NNANDS_nand2	18	6,96	9,225	64,206
NORS_nor3	16	7,96	6,46	51,4216
NNORS_nor3	18	7,96	8,64	68,77
ANDS	24	15,03	21,79	327,5

3.6. Construcción del decodificador de fila con enable

3.6.1. Decodificador de 4 por 16 bits

El primer decodificador, 4x16, se construye con tres etapas basadas en los sub-bloques BASE_NOR, NANDS/NNANS_nand1 y NORS/NNORS_nor2. En la figura 3.34 se puede observar la interconexión y la estructura del decodificador implementado (primero su diagrama preliminar y luego el trazado final).

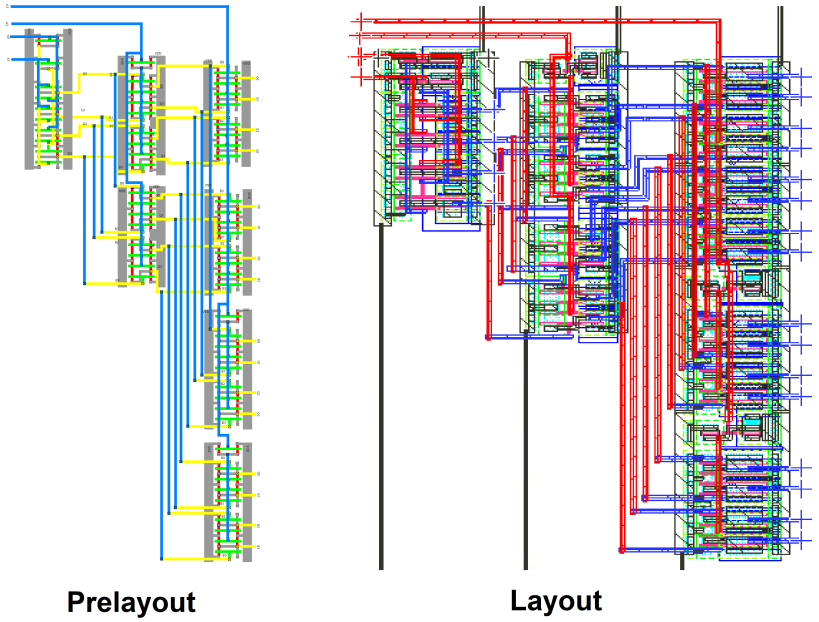
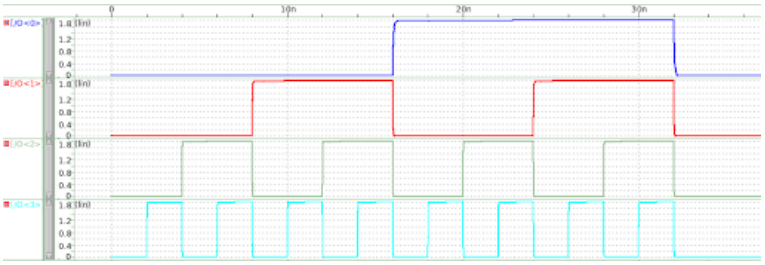
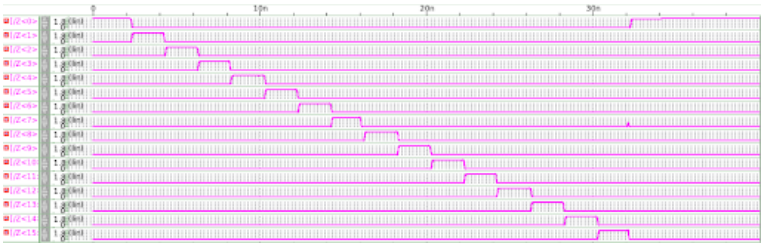


Figura 3.34: Vistas de trazado preliminar y final de decodificador 4x16.

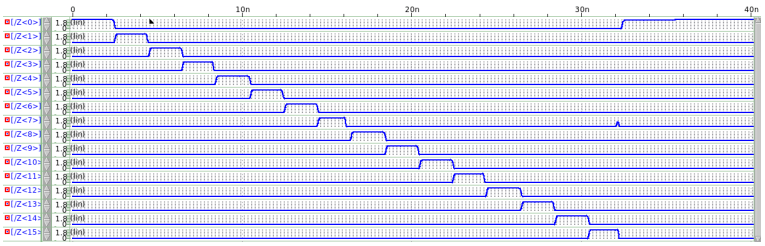
A partir del trazado se generan simulaciones con parásitas extraídas en HSPICE. Se realiza una verificación exhaustiva, con un contador recorriendo todas las posibles direcciones. La figura 3.35 resume los resultado obtenidos.



(a) Señales de dirección.



(b) Respuesta del esquemático.



(c) Respuesta del *layout*.

Figura 3.35: Respuesta temporal de una simulación exhaustiva SPICE del decodificador. No se aprecian fallas lógicas.

Las simulaciones de este bloque ofrecen un comportamiento esperado.

3.6.2. Decodificador de 5 por 32 bits

Una vez validado el comportamiento del decodificador de 4x16 bits, se agrega la siguiente etapa compuesta por NANDS/NNANDS_nand2 para llegar al decodificador de 5x32 bits. Los diagramas previos y de trazado final se muestran en las figuras 3.36.

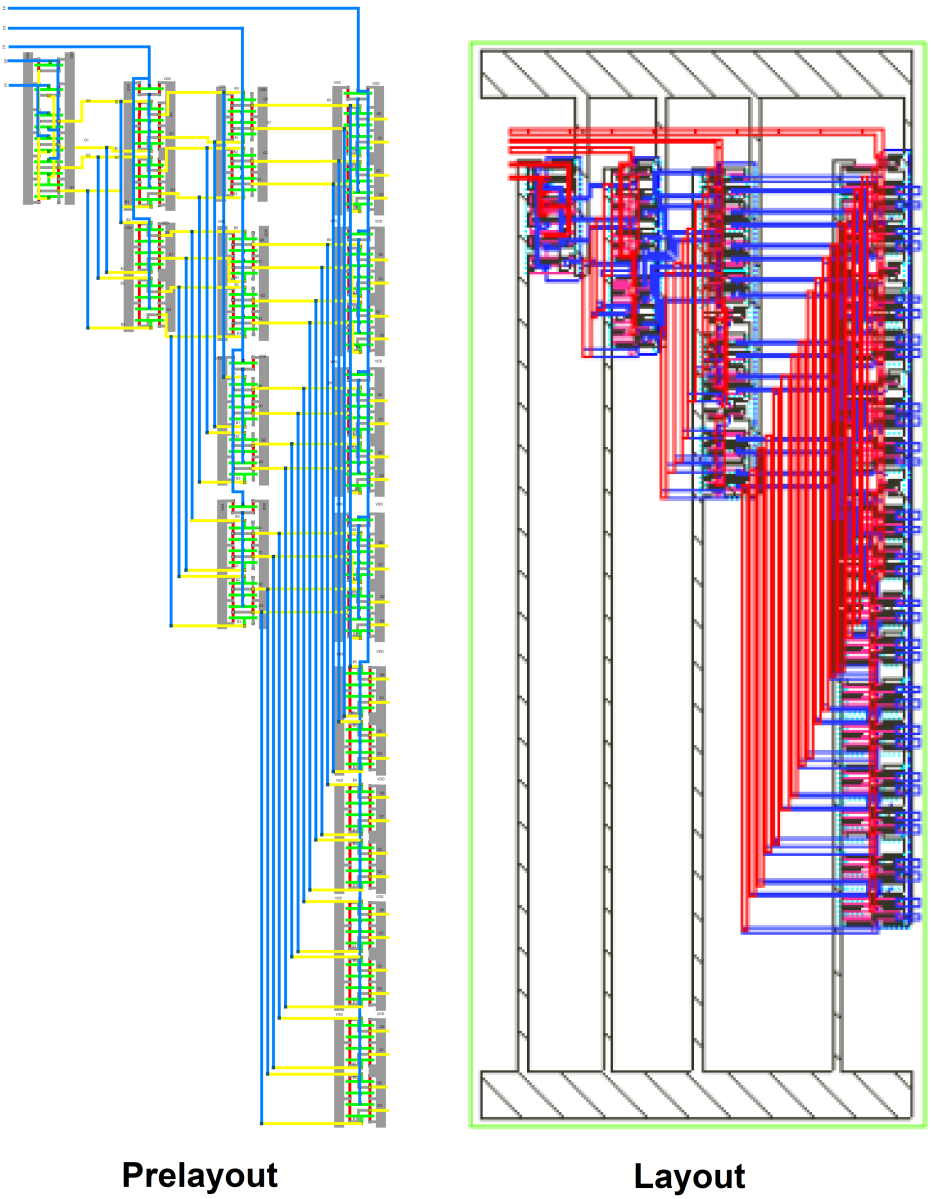
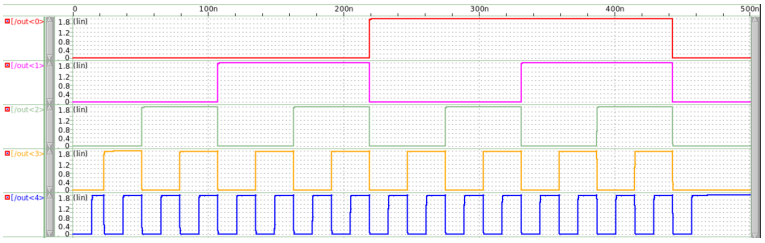
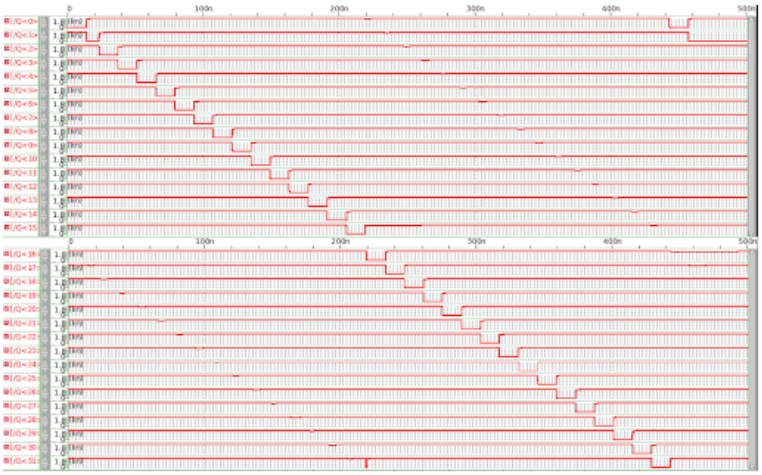


Figura 3.36: Trazado preliminar y final del decodificacor de 5x32.

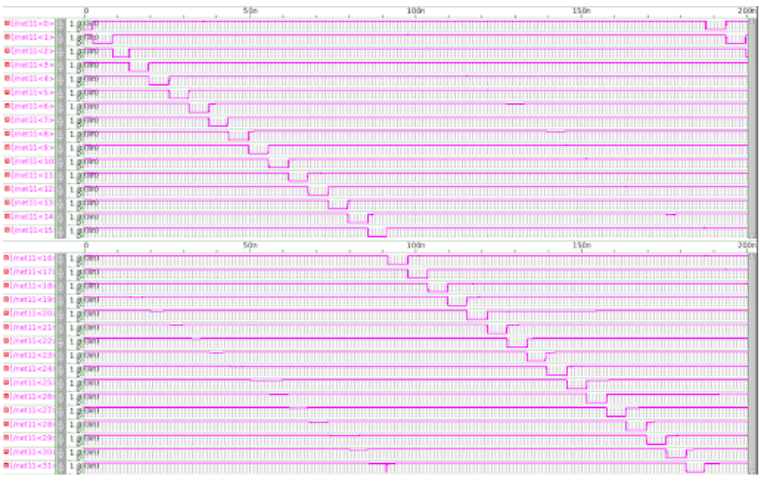
Se realizó el mismo proceso de verificación usado para el decodificador anterior.



(a) Señales de dirección.



(b) Respuesta del esquemático.



(c) Respuesta del *layout*.

Figura 3.37: Resultados de simulación HSPICE en esquemático y post-layout del decodificador 5x32.

El funcionamiento es correcto, pero puede notarse un *glitch* en el bit 31 en ambas simulaciones, producto de los retardos de los caminos críticos. Este sin embargo no ofrece problemas si el temporizado de acceso a la SRAM es correcto.

3.6.3. Decodificador de 6 por 64 bits con enable

El proceso de desarrollo de este decodificador se monta sobre los dos diseños anteriores, por lo que se ofrecen solo los resultados finales de trazado.

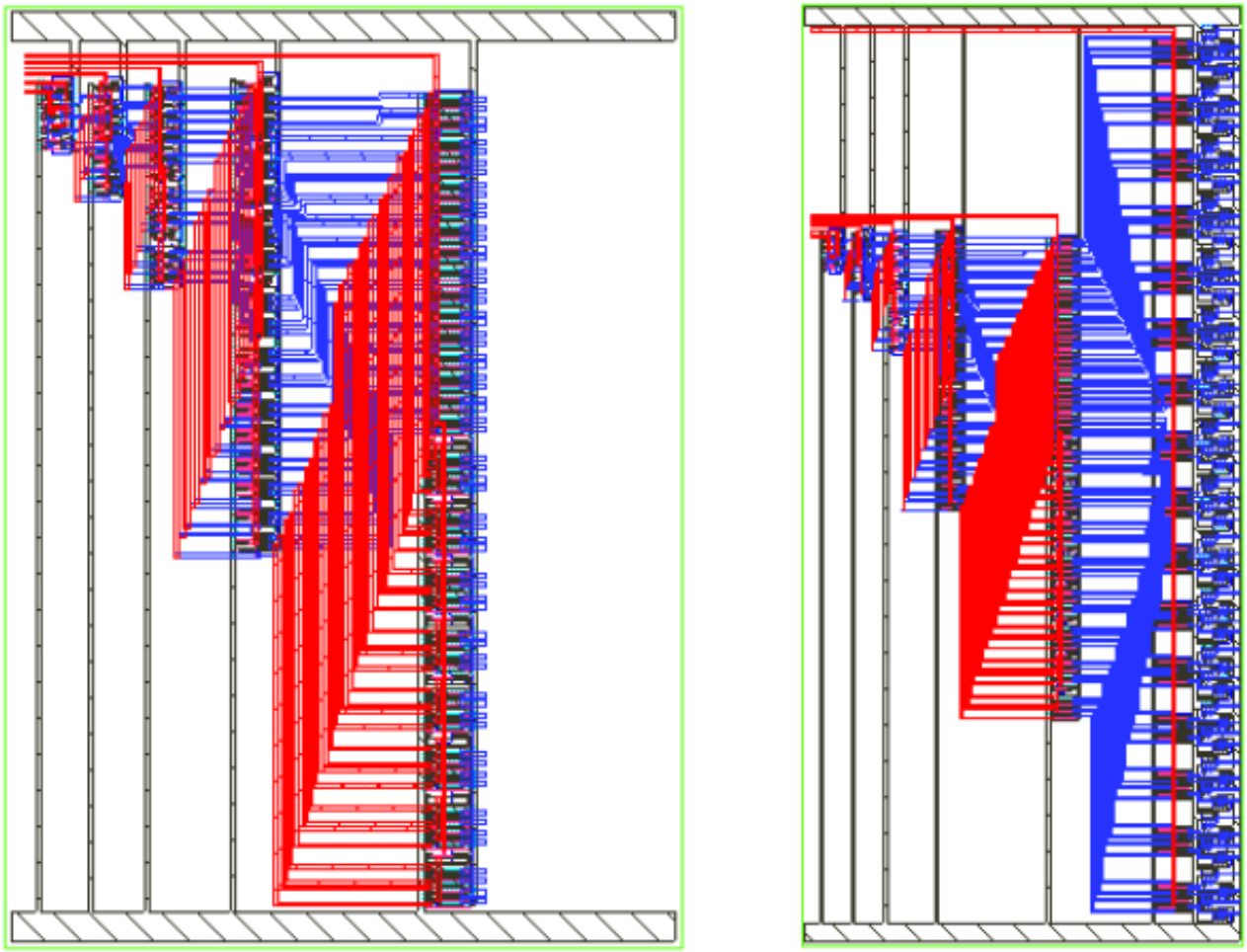


Figura 3.38: Trazados del decodificador de 6x64 bits con y sin etapa de *enable*.

Es de notar en ambos circuitos el exceso de área desperdiciada. Ello obliga a reubicar los módulos para mejor aprovechamiento del espacio.

Para la reubicación de los sub-bloques, se crean modelos a escala que cumpla aproximadamente con el área obtenida en la tabla 3.11, para así tener una vista general de la

nueva posición y su relación con los demás bloques de una manera más cercana al resultado a esperar en el trazado. Para lograr lo anterior se mueven los bloques de tal forma que se aproveche el espacio cercano a los rieles y se reduzca el ancho total del decodificador.

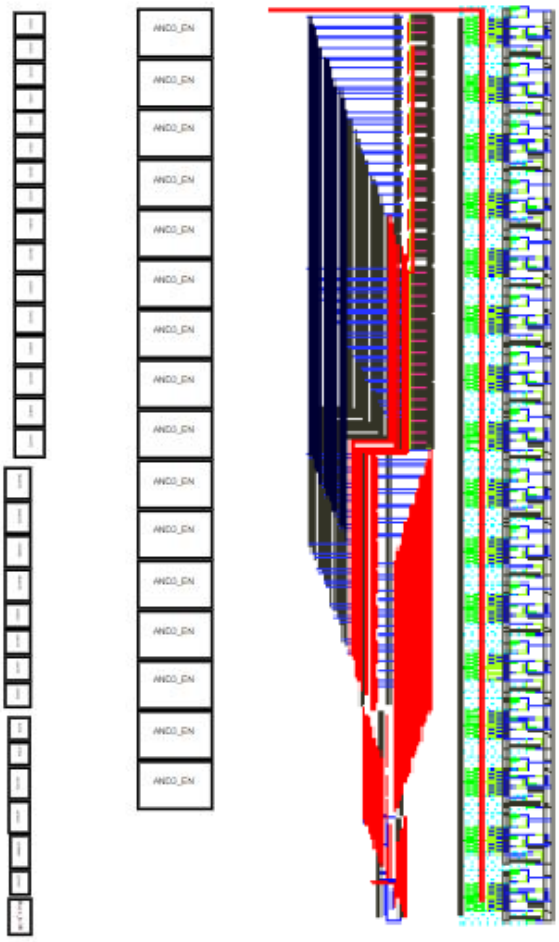


Figura 3.39: Diagrama de la estructura de sub-bloques y trazado optimizando .

Se permiten ciertos espacios pensando en el cableado necesario para interconectar las etapas. Luego de desplazar y cablear con las nuevas posiciones se obtiene el trazado de la derecha, donde se ve que era necesario tomar en cuenta las separaciones entre los sub-bloques. A simple vista el trazado anterior se ve más compacto y se ajusta mejor a la última etapa, también se puede observar que estas dos etapas se encuentran aún desconectadas, debido a que se procederá a mejorar el trazado de la última etapa, para posteriormente juntar lo más posible estos circuitos.

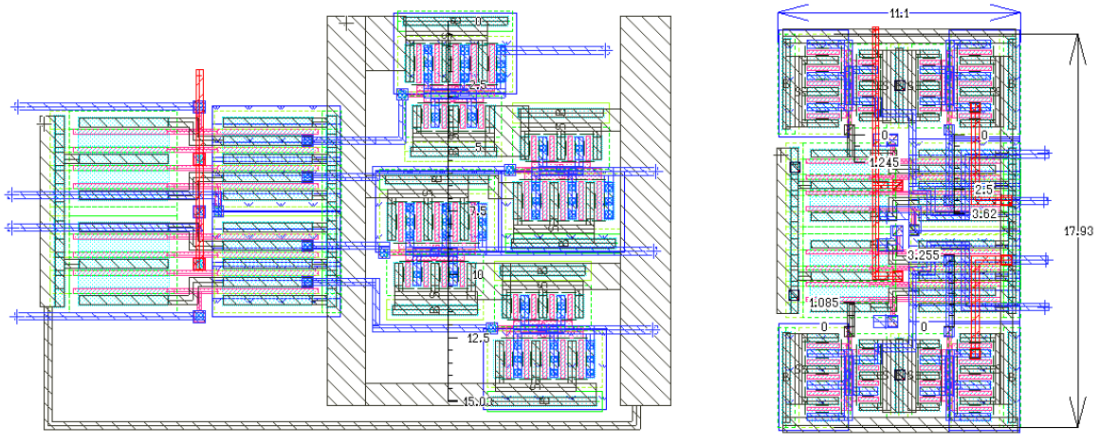


Figura 3.40: Trazados de la etapa de *enable*.

En el diseño inicial se pueden notar los hoyos en el diseño que causan desperdicio, por lo cual se reordena hasta llegar al trazado optimizado de la derecha. Las dimensiones de los trazados anteriores se presentan en la la tabla 3.12.

Tabla 3.12: Resultado de las mejoras en el trazado del circuito *enable*.

Trazados	**Alto**(μm)	**Largo**(μm)	**Área** (μm^2)
Diseño inicial	15,81	22,74	359,5
Diseño optimizado	17,93	11,1	199

Según la tabla 3.12 el área inicial se redujo aproximadamente 160 μm^2, lo que a final mejora aún más el área total de diseño. En la figura se ofrece un nuevo trazado ya incorporando el circuito de *enable* mejorado.

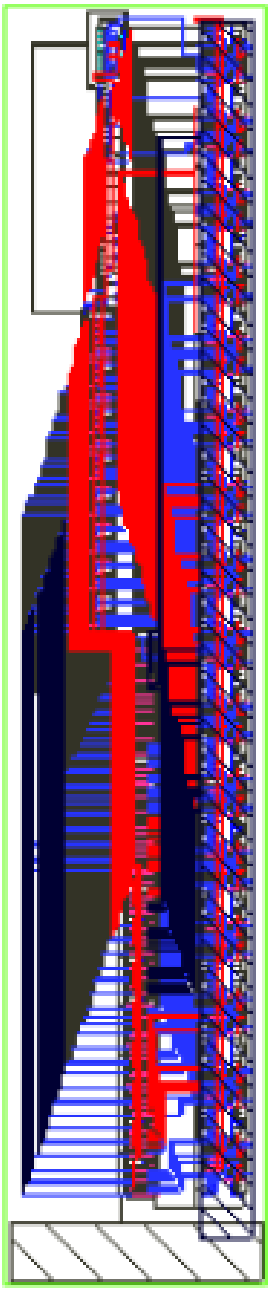


Figura 3.41: Trazado del decodificador de 6x64 bits con *enable* optimizado.

Es claro el mejor aprovechamiento de las áreas en blanco.

Tabla 3.13: Contraste entre el trazado inicial y el mejorado en el decodificador de 6×64 bits.

Layouts	**Ancho** (um)	**Largo** (um)	**Área** (um^2)
Diseño inicial	255	117	29835
Diseño optimizado	296,6	57	16906,7

Nóstese la mejora alcanzada en casi 13mil μm^2 en el área.

Para el sub-bloque de los *latches* se proponen dos diseños, ambos basados en el principio de realimentación, que se muestran en la figura 3.42.

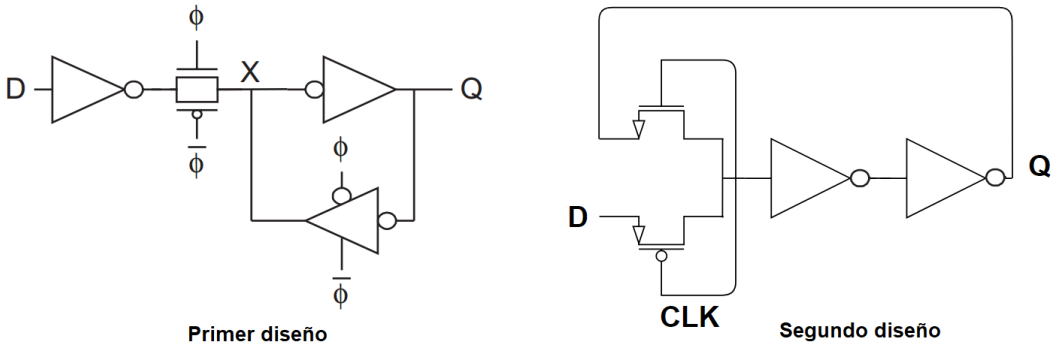


Figura 3.42: Diseños propuestos para los *latches*.

Como una de las principales características que se buscan en los diseños es el área que abarcan los trazados, y la cantidad de transistores es directamente proporcional a ésta, es claro que el primer diseño posee una mayor cantidad de transistores. Con la compuerta de paso, el inversor triestado y los dos inversores se suman un total de diez transistores, mientras que para el segundo diseño tenemos solo seis transistores en total. El ahorro general sería de 24 transistores, con esta gran ventaja se selecciona el diseño de a implementar en trazado, además como el diseño solo tiene un par de inversores y un par de transistores, se pasa directamente al desarrollo del trazado.

Como estrategia se pretende renovar las direcciones en el ciclo negativo del reloj con el fin de que los bits de dirección se propaguen con antelación y, una vez que el reloj conmute, los *latches* sostengan esta dirección, lo que permite que la dirección ya se encuentre decodificada antes del flanco del reloj. Para cumplir con el objetivo anterior el transistor de la entrada debe ser tipo P y el que permite la realimentación debe ser tipo N.

Una situación a considerar es que debido a la tensión de activación de los transistores, hay caídas de tensión en el transistor. En el caso de los 0 lógicos estos se degradan al cruzar los transistores PMOS y en el caso de los 1 lógicos se degradan al cruzar los NMOS. Por esta razón se mantiene al mismo potencial la terminal del cuerpo con la del surtidor para tener efecto de cuerpo y bajar esta tensión de activación del transistor. Adicionalmente la realización de esta conexión es físicamente posible debido a que la tinas de NWELL del PMOS son separables, a diferencia del NMOS.

La carga más grande que ejerce el decodificador sobre los *latches* es de la entrada $A5$, por que este es el nodo con más compuertas y con transistores más grandes. Por ello que se replicará este *latch* para las demás entradas de direcciones, ya que si este diseño funciona sobre esta entrada, funcionará para las otras entradas. La entrada $A5$ maneja 16 bloques

de cuatro compuertas NOR de la etapa de **nor3** del decodificador, para un total de 64 compuertas NOR iguales.

Cuando se realizó el dimensionamiento del decodificador, se había determinado que el valor de la capacitancia de entrada de las compuertas NOR de la etapa NOR3 era de 25,54 C, ahora la capacitancia total que debe manejar el *latch* es de 25.54C por 64 aproximadamente 1634 C.

Realizando simulaciones a tamaño mínimo en los transistores de paso, primer inversor mínimo y segundo inversor de dos veces el primero conectado a la carga equivalente calculada en capacitancia, se obtienen los resultados de la figura 3.43.

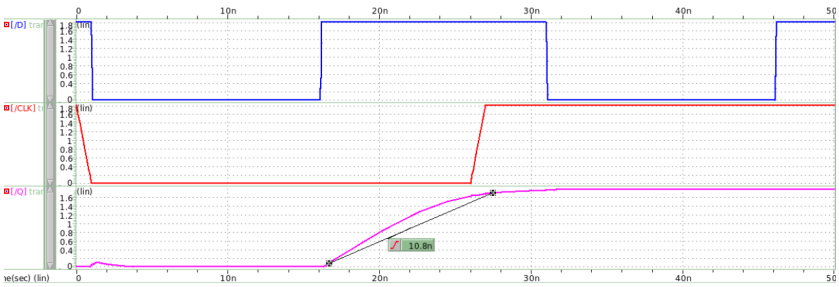


Figura 3.43: Simulación del segundo diseño con carga.

La señal azul son los datos en la entrada *D* del *latch*, la roja es la señal de reloj y la roja la salida *Q*. Los efectos de la carga son bastante notables en el tiempo de subida de la salida del *latch* y su tiempo de propagación es de alrededor de 4,4 ns, esto supera las limitaciones que debe cumplir este sub-bloque. Adicionalmente se impacta directamente el retardo general, su tiempo de *setup* y consecuentemente es necesario esperar más tiempo para que el decodificador funcione. Es necesario distribuir mejor la carga al costo de más etapas, colocando un inversor al inicio que maneje los datos de entrada y uno al final que pueda restituir el dato de salida y que mantenga el dato original. Se propone la solución de la figura 3.44.

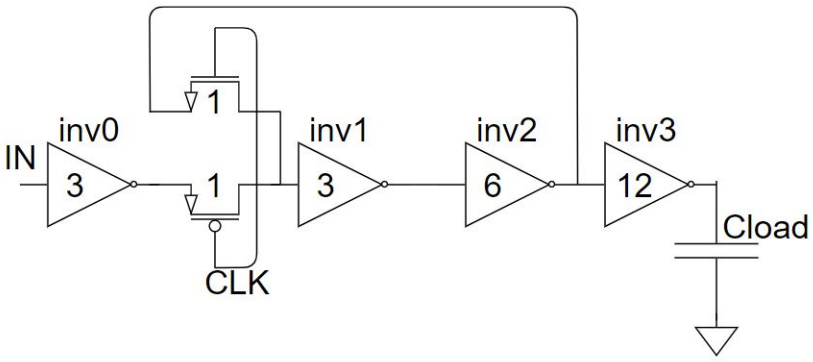


Figura 3.44: Diseño del *latch* con cadenas de *buffers*.

En una nueva simulación, es posible apreciar, en la figura 3.45, la mejora en le tiempo tanto de subida como de propagación.

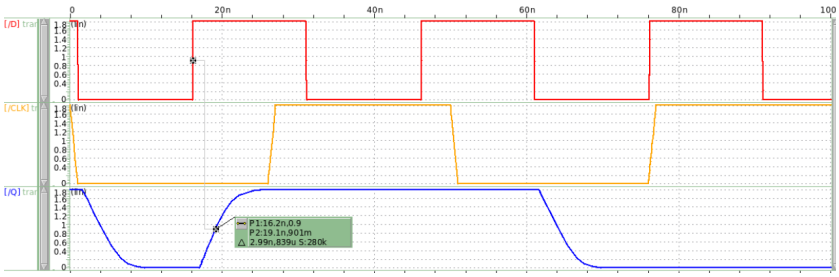


Figura 3.45: Simulación con nueva cadena de *latch* propuesto.

El caso particular simulado muestra un retardo de 2,9 ns, el cual cumple con las especificaciones planteadas del proyecto. La figura 3.46 ofrece el trazado final del *latch* seguido de una cadena de *buffers*.

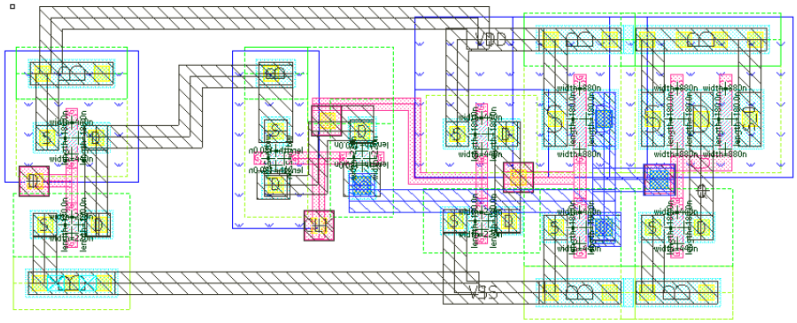


Figura 3.46: Trazado del *latch* con cadena de *buffers*.

El trazado del *latch* podría optimizarse en términos de área, pero cabe recordar que los pozos de NWELL para distintos potenciales deben cumplir una distancia mínima. El bloque final apilado de *latches* se muestra en la figura 3.47.

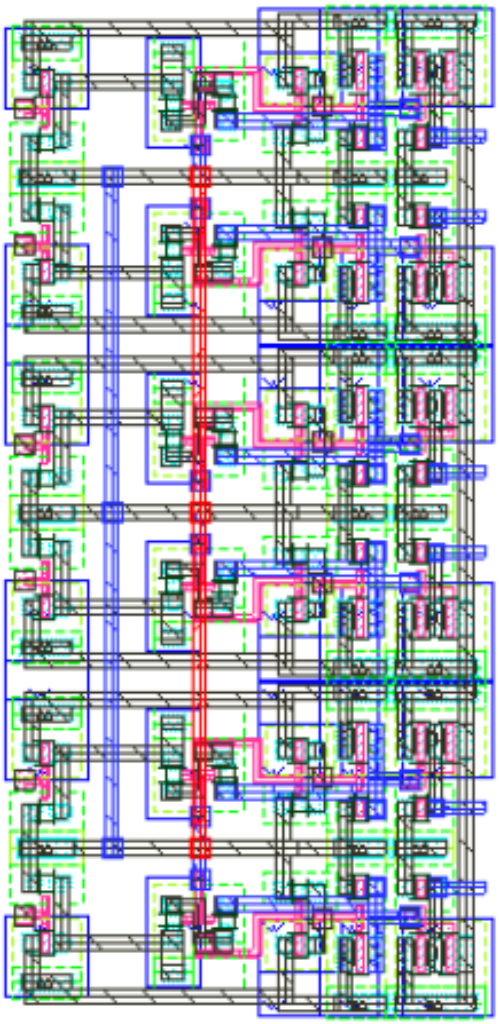


Figura 3.47: *Layout* del sub-bloque de *latches*.

Con todos los trazados de los sub-bloques pertenecientes a los circuitos del decodificador de fila listos, solo es necesario interconectarlos entre sí, en integrar el módulo de fila completo al arreglo de memoria para realizar pruebas de verificación funcional y caracterización de retardos de los bloques individuales con las cargas parásitas.

Al añadir el sub-bloque de *latches* al diseño, el área total se mantiene igual por que se aprovecha la esquina superior izquierda del decodificador, que ya se había tomado en cuenta en la medición de la figura 3.48.

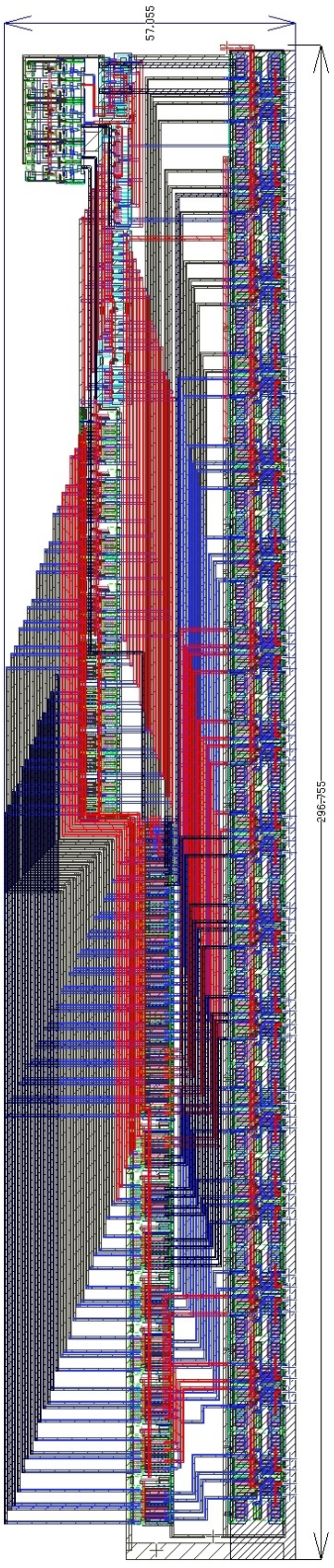


Figura 3.48: Trazado final del decodificador completo.

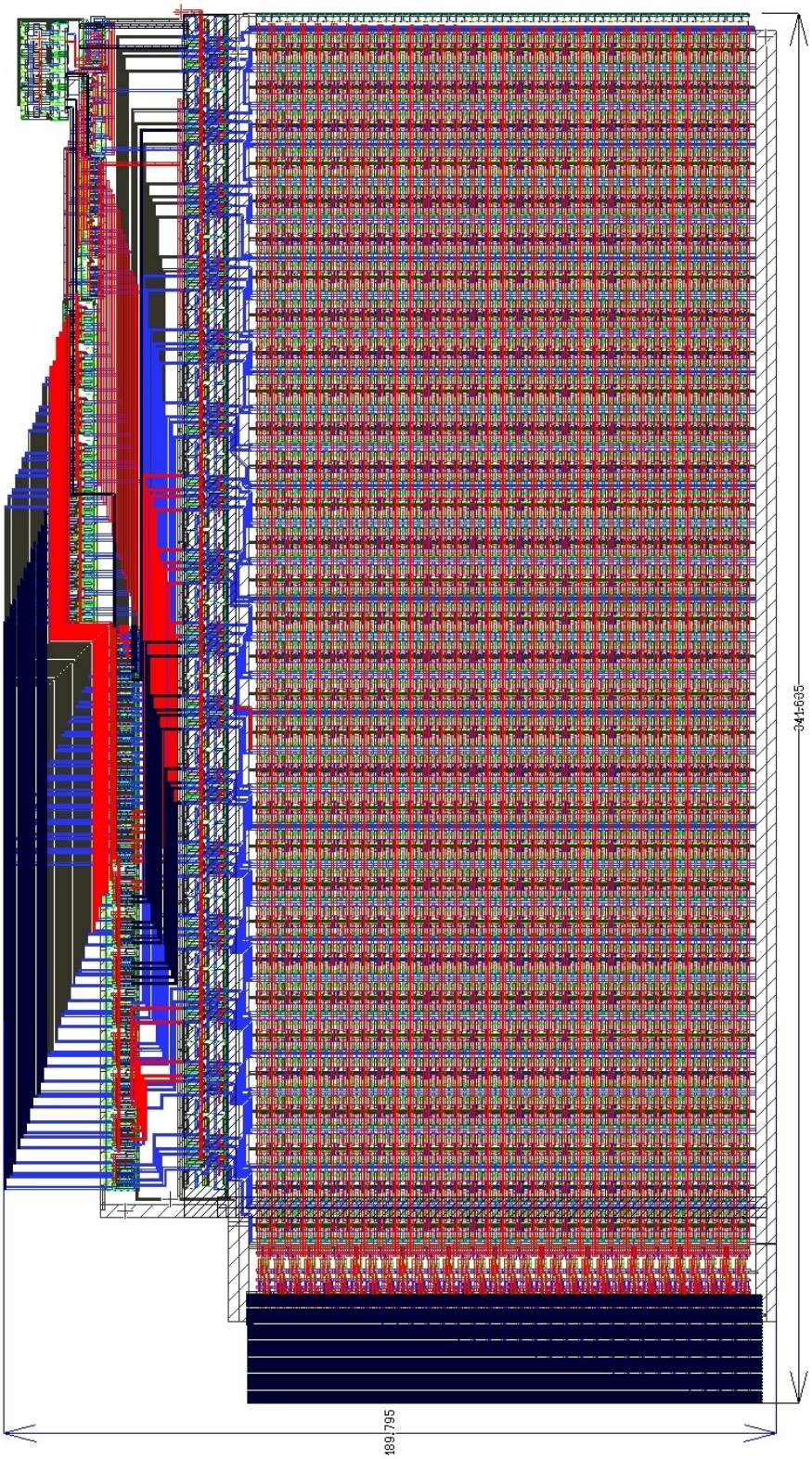


Figura 3.49: Trazado final de memoria completa.

Sobre la extracción de la figura 3.49 se realizan las pruebas de funcionamiento.

3.8.1. Escritura y lectura de la memoria

Con el fin de comprobar que el decodificador cumple su papel una vez integrado a la memoria, se realiza una prueba con frecuencia de reloj de 20 MHz para escribir un dato y leer ese mismo dato en el siguiente ciclo en diferentes posiciones de memoria.

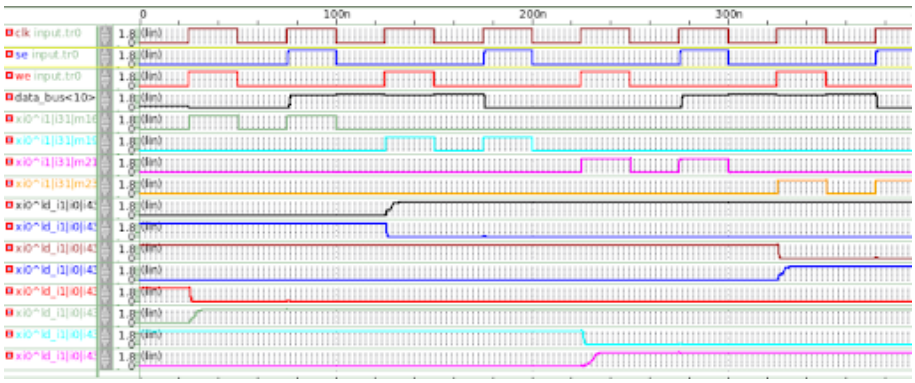


Figura 3.50: Simulación de escrituras y lecturas.

En la simulación anterior los datos de todos los 32 bits son iguales, por lo que solo se muestra la señal de un bit del bus de datos de la memoria (en este caso la señal número 4 en color negro). Las señales 5, 6, 7 y 8 de color gris, azul claro, morado y naranja respectivamente, son las *wordlines* a las que se está realizando la escritura y seguidamente la lectura. Como se observa, en sincronía con el primer acceso se sobreescriben bits de las *wordline*, que son las últimas ocho señales y, debido a la construcción del circuito de lectura, los datos leídos son los complementos de como se cargaron en el bus, por esta razón el valor del dato en el bus se invierte con cada acceso.

3.8.2. Potencia del decodificador

Para obtener la potencia promedio aproximada de únicamente el módulo de decodificación de fila, se realiza una simulación separada de la memoria con la extracción de parásitos del decodificador, conectado a un banco de capacitores con el valor calculado de la capacitancia de las *wordlines* de la memoria, conmutando a una frecuencia 20 MHz con diferentes duraciones de tiempo de simulación. Es necesario separar estos módulos ya que es imposible observar la corriente individual del módulo una vez integrado a los rieles de alimentación del arreglo de memoria.

Tabla 3.14: Potencia promedio aproximada del decodificador.

Tiempo de simulación (us)	I_{supply} (uA)	**Potencia** (uW)
1	30,41	54,738
4	30,71	55,278
7	30,97	55,746
10	31,11	55,998
13	31,15	56,07
16	31,06	55,908
64	31,1	55,99

Según la tabla anterior la potencia promedio consumida solo por el módulo de fila de la memoria es alrededor de los 56 uW.

3.8.3. Retardos del sub-bloque de latches

Para obtener el peor de los tiempos de *setup* se realizan las simulaciones sobre el bit más significativo de dirección del decodificador, que es el que posee más carga. También en este caso se realizaron las simulaciones para las transiciones en donde se captura un 1, porque son los estados que más duran en cargarse.

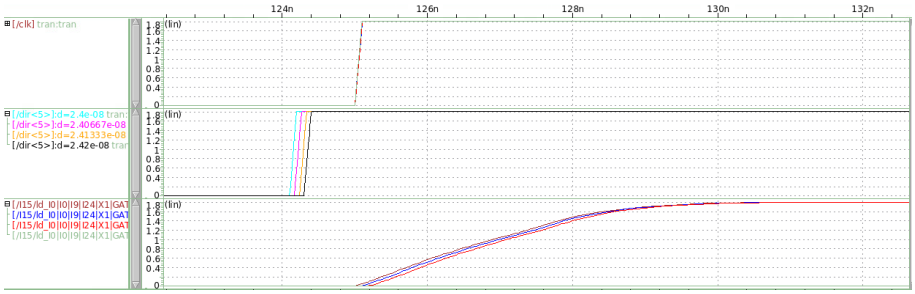


Figura 3.51: Simulación para determinar tiempo de *setup* en 1 lógicos.

Para obtener el tiempo de *setup* del *latch* se realiza una simulación de tipo barrido hasta que la algunas de las señales más cercanas al flanco de reloj no se capture en el *latch*, como se puede observar en la figura 3.51, la última de las señales no se captura, por lo que se revisa la señal anterior, que si es capturada, ésta cuenta con un retado de retardo respecto al flanco de reloj de aproximadamente 770 ps, este valor implica que si se trabaja con datos estables deseados a almacenar 770 ps antes del flanco, todos los *latches* son capaces de capturar las direcciones.

Para obtener el peor tiempo de propagación de las direcciones a través del *latch*, igualmente se simula sobre el bit más significativo. La señal de reloj se mantiene en 0 para permitir el paso de las direcciones de D hasta Q, se muestran los resultados en la siguiente tabla.

Tabla 3.15: Tiempos de propagación del *latch* más significativo.

t_{pdr} (ns)	t_{pdf} (ns)
2,59	2,2
2,61	2,19
2,6	2,19
2,61	2,25
2,6	2,19
2,59	2,19
2,61	2,19
2,6	2,19
2,65	2,19
2,59	2,19

Según la tabla anterior el tiempo típico de propagación para los flancos de subida es de 2,6 ns y para los flancos negativos es de 2,19 ns, para un retardo promedio aproximado de 2,4 ns. Este valor difiere del la simulación *prelayout* obtenida en la figura 3.45, lo que indica una sobreestimación a beneficio del retardo.

3.8.4. Retardos del sub-bloque de enable

Para obtener los tiempos de propagación del circuito del *enable*, se mantiene la señal de **enable** en 1 y se permite pasar las señales decodificadas del sub-bloque anterior. Se muestran los resultados en la siguiente tabla.

Tabla 3.16: Tiempos de propagación de sub-bloque de *enable*.

t_{pdr} (ps)	t_{pdf} (ps)
194	238
199	239
198	229
298	228
201	241
206	228
200	236
204	236

Los datos anteriores fueron tomados de una simulación donde se decodificó de arriba hacia abajo las salidas que van hacia la memoria. Los tiempos de propagación promedio son de 200 ps y 234 ps para los flancos se subida y bajada respectivamente, lo que lleva a un total aproximado de retardo de 217 ps.

3.8.5. Retardos del sub-bloque del decodificador

Para la medición de estos tiempos de propagación, se mide en las diferentes salidas del decodificador antes de la etapa de *enable*, generando las señales de las direcciones en secuencias ordenadas en su entrada.

Tabla 3.17: Tiempos de propagación de sub-bloque del decodificador.

t_{pdr} (ns)	t_{pdf} (ns)
1,43	1,3
1,74	1,34
1,37	1,42
1,09	1,18
1,43	1,3
1,74	1,34
1,38	1,3
1,74	1,42

Con la tabla anterior se puede obtener el tiempo de retardo promedio para los flancos de subida, que es de 1,49 ns y para el flanco de bajada es de 1,31 ns, y el promedio aproximado general para este sub-bloque es de 1,4 ns.

3.8.6. Retardo de propagación del módulo de fila completo

Para medir los tiempos de propagación sobre todo el bloque de fila se mantienen las señales de *enable* siempre en 1 y el reloj siempre en 0, luego se mide entre las diferentes transiciones de las señales de dirección en la entrada hasta las salidas decodificadas.

Tabla 3.18: Tiempos de propagación del módulo de fila completo.

t_{pdr} (ns)	t_{pdf} (ns)
2,57	2,47
2,53	2,16
2,5	2,58
1,86	1,99
2,55	2,47
2,52	2,16
2,53	2,59
2,57	2,47

Los valores promedios para los tiempos de propagación son de 2,45 ns y 2,36 ns para t_{pdr} y t_{pdf} respectivamente, lo que da un total en el tiempo de retardo de todo el diseño de 2,4 ns. Cabe destacar que si se sumasen los tiempos de propagación promedio de los 3 sub-bloques sería aproximadamente de 4 ns, a con una diferencia de 1,6 ns medido anteriormente sobre todo el bloque de fila.

A manera de resumen, se implementa todo el diseño fabricable del decodificado de fila a una alimentación de 1,8 V. Simulado con una frecuencia de reloj de 20 MHz, usando las herramientas de EDA de Synopsys Custom Compiler para la tecnología CMOS de 180 nm. Contrastando con los objetivos iniciales y limitaciones del proyecto con los resultados finales del proyecto, se muestra la tabla comparativa 3.19.

Tabla 3.19: Resultados finales

	Objetivo	**Resultado**
Altura (μm)	300	296,6
Decodificador (ns)	5	1,4
Latches (ns)	3	2,4
Buffers (ns)	2	0,217

Todos los objetivos se cumplen satisfactoriamente para los resultados obtenidos con el diseño propuesto, por último se muestra en la tabla 3.20 una comparación entre el retardo calculado desde el flanco de reloj, el cual está conectado a la señal de *enable* hasta capacitores con el valor calculado de las *wordlines* contra el retardo del diseño una vez integrado a la memoria en una simulación *postlayout*.

Tabla 3.20: Resultados finales

$enable - wl$	**Calculado**	**Simulación**
Retardo (ps)	170	183,75

La diferencia entre los retardos promedios es de aproximadamente 13 ps, por lo que las consideraciones realizadas son bastantes buenas respecto al resultado esperado.

4 Conclusiones y recomendaciones

4.1. Conclusiones

Se ha descrito el proceso de diseño y desarrollo de los circuitos del decodificador de fila necesarios para el funcionamiento de la memoria SRAM sobre una tecnología CMOS de 180 nm. El desarrollo se llevó a cabo con las herramientas EDA Synopsys de Custom Compiler.

Se ha verificado asimismo, a través de diferentes simulaciones mediante HSPICE, el funcionamiento adecuado bajo las especificaciones propuestas del proyecto tanto de sus sub-bloques de fila individualmente como de forma conjunta con el arreglo de memoria inicial.

Por medio de diferentes técnicas para la creación de trazados para los circuitos diseñados, y con ayuda del *set* de reglas se logran obtener diseños compactos y trazados a medida de tal manera que se acoplaron con las dimensiones del arreglo de memoria. Estos circuitos pueden reutlizarse para futuros proyectos donde sea requerido el crecimiento tanto en la cantidad de *wordlines* o espacios de memoria como columnas de bits aún mayores, usando como base los sub-bloques propuestos. El trabajo aquí desarrollado para decodificación de fila arreglos de memoria, puede usarse también para controlar decodificadores de columna.

Existen diferencias entre las aproximaciones hechas cuando se compara entre las simulaciones con modelos capacitivos a las simulaciones *postlayout* parásitas. Estas diferencias se justifican por el uso de un modelo pesimista para los cálculos iniciales que arrancaron los diseños. Adicionalmente, nótese que no se incorporaron en los análisis iniciales, ningún efecto resistivo o capacitivo debido al alambrado de la tecnología.

4.2. Recomendaciones

Se recomienda estudiar la optimización de tamaño de los sub-bloques de predecodificación, incluyendo si es posible el estudio del costo de alambrado, por medio de algún algoritmo heurístico, tal como los descritos en [16, 17, 18].

Con el fin de optimizar aún más área, se puede trabajar sobre la etapa de *enable*, reduciendo los anchos de los transistores, ya que estos son los más grandes. Con el estudio adecuado se puede sacrificar velocidad sobre el arreglo de memoria, mientras que el arreglo de memoria sea capaz de realizar las escrituras y las lecturas sin fallas.

Se recomienda también apuntar en la dirección de crear una biblioteca de celdas estándar ajustadas a las dimensiones del arreglo de SRAM. Ello permitiría la generación de un procedimiento semi automatizado de generación de módulos de SRAM para futuros proyectos.

5 Bibliografía

[1] ABC de dispositivos médicos. (2018). Bogotá D.C, p.21.

[2] Azizi-Mazreah, A., T. Manzuri Shalmani, M., Barati, H. and Barati, A. (2008). Delay and Energy Consumption Analysis of Conventional SRAM. International Journal of Electrical and Computer Engineering.

[3] Baker, J. (2018). Circuit Design, Layout, and Simulation. 3rd ed. Canada: Jhon Wiley and Sons, Inc.

[4] Bhaskar, A. (2017). Design and Analysis of Low Power SRAM Cells. Vellore, India.

[5] C. Salazar-García, R. Castro-González, and A. Chacón-Rodríguez, "RISC-V based sound classifier intended for acoustic surveillance in protected natural environments," presented at the Circuits & Systems (LASCAS), 2017 IEEE 8th Latin American Symposium on, 2017, pp. 1–4.

[6] C. Mead y L. Conway, Introduction to VLSI Systems, Reading, MA: Addison-Wesley, 1980.

[7] CMOS VLSI Design: A Circuits and Systems Perspective. ,4 edition.Boston: Addison-Wesley,2010.

[8] E. Salman, A. Dasdan, F. Taraporevala, K. Kucukcakar, and E.G. Friedman. Pessimism Reduction In Static Timing Analysis Using Interdependent Setup and Hold Times. Proceedings of the 7th International Symposium on Quality Electronic Design, page 6, March 2006.

[9] Fda.gov. (2018). Medical Devices. Disponible en: https://www.fda.gov/MedicalDevices/default.htm [Accesado 17 May 2018].

[10] H.Bakoglu, Circuits, Interconnections, and Packaging for VLSI, Addison-Wesley publishing company, 1990.

[11] I. Brzozowski, Ł. Zachara, A. Kos, "Design method of compact n-to-2n decoders," Intl Journal of Electronics and Telecomunications, vol. 59, no. 4, December 2013, pp. 405-413

[12] I. Brzozowski, P. Dziurdzia, A. Kos, "Design and Analysis of Multi-Level n-to-2n Decoders in CMOS Technology," AGH-University of Science and Technology, Mickiewicza 30, 30-059 Kraków, Poland

[13] I.E. Sutherland, "Micropipelines", Communications of the ACM, Vol. 32 No. 6, June 1989, pp. 720-738.

[14] Kang, S. y Leblebici, Y. (2003). CMOS Digital Integrated Circuits: Analysis and Design. 3rd ed. McGraw-Hill, p.22.

[15] L. Heller, W. Griffin, J. Davis, and N. Thoma, "Cascode voltage switch logic: a differential CMOS logic family," Proc. IEEE Intl. Solid-State Circuits Conf., 1984, pp. 16–17.

[16] Pereira-Arroyo, R.; Nicaragua-Guzmán, F.; Chacón-Rodríguez, A.; , "Design of an Operational Transconductance Amplifier applying multiobjective optimization,.ªrgentine School of Micro-Nanoelectronics Technology and Applications (EAMTA), 2010 , vol., no., pp.12-17, 1-9 Oct. 2010.

[17] R. Pereira-Arroyo, A. Chacon-Rodríguez. "Diseño de una biblioteca de compuertas MCML utilizando un algoritmo genético y de optimización multiobjetivo". Tecnología en Marcha, vol. 27, no. 4, pp. 41-48 , Oct.-Dic. 2014.

[18] R. Pereira-Arroyo, R. Molina-Robles, A. Chacon-Rodríguez. "Diseño de un amplificador operacional de transconductancia aplicando técnicas de optimización multiobjetivo". Tecnología en Marcha, vol. 27, no. 1, pp. 2-12 , Ene.-Mar. 2014.

[19] Salazar-Garcia, C.; Alfaro-Hidalgo, L.; Carvajal-Delgado, M.; Montero-Aragon, J.; Castro-Gonzalez, R.; Rodriguez, J.A.; Chacon-Rodriguez, A.; Alvarado-Moya, P., "Digital integrated circuit implementation of an identification stage for the detection of illegal hunting and logging,ïn Circuits & Systems (LASCAS), 2015 IEEE 6th Latin American Symposium on , pp.1-4, 24-27 Feb. 2015

[20] Tocci, Widmer y Moss. Sistemas Digitales Principios y Aplicaciones. Pearson Prentice Hall: 10º Ed. 2007.

[21] W. Elmore, "The transient response of damped linear networks with particular regard to wideband amplifiers," J. Applied Physics, vol. 19, no. 1, Jan. 1948, pp. 55–63.

[22] Y. Tsividis, Operation and Modeling of the MOS Transistor, 2nd ed., Boston: McGraw-Hill, 1999